aws

Guía para desarrolladores

Amazon GameLift

Amazon GameLift: Guía para desarrolladores

Table of Contents

¿Qué es Amazon GameLift?

Puede utilizar Amazon GameLift para implementar, operar y escalar servidores dedicados y de bajo costo en la nube para juegos multijugador basados en sesiones. Como se ha desarrollado en la infraestructura informática global de AWS, Amazon GameLift le permite ofrecer servidores de juegos de alto rendimiento y fiabilidad a bajo costo y, al mismo tiempo, escalar de forma dinámica el uso de recursos para adaptarse a las necesidades de los jugadores de todo el mundo.

Usos de Amazon GameLift

Amazon GameLift admite estos casos de uso y más:

- Utilice sus propios servidores de juegos multijugador personalizados o emplee servidores de Realtime listos para usar para alojar sus juegos.
- Ejecute recursos de alojamiento de bajo costo mediante instancias de spot de Amazon Elastic Compute Cloud (Amazon EC2).
- Escale automáticamente la cantidad de recursos de alojamiento que necesita su juego en función del uso.
- Administre sus recursos informáticos de Amazon EC2 en un solo lugar mediante Amazon GameLift FleetIQ.
- Empareje jugadores en juegos multijugador con Amazon GameLift FlexMatch.
- Pruebe de forma iterativa las compilaciones de sus servidores y clientes de juegos con Amazon GameLift. Anywhere.
- Utilice su propio hardware y administre todo desde un solo lugar con Amazon GameLift Anywhere.

> **Tip**
>
> Para probar el alojamiento del servidor de juegos de Amazon GameLift, consulte Introducción a Amazon GameLift.

Introducción a las soluciones de Amazon GameLift

Soluciones de Amazon GameLift para desarrolladores de juegos

- Alojamiento de Amazon GameLift para servidores personalizados
- Alojamiento de Amazon GameLift con Realtime Servers
- Amazon GameLift FleetIQ para alojamiento en Amazon EC2
- Amazon GameLift FlexMatch para emparejamiento
- Alojamiento de hardware de Amazon GameLift Anywhere

Alojamiento de Amazon GameLift para servidores personalizados

Amazon GameLift reemplaza el trabajo necesario para alojar sus propios servidores de juegos personalizados. Las capacidades de escalado automático le ayudan a evitar tener que pagar por más recursos de los que necesita. El escalado automático también le ayuda a garantizar que siempre tenga juegos disponibles para que los nuevos jugadores se unan con un tiempo de espera mínimo.

Para obtener más información sobre el alojamiento de Amazon GameLift, consulte Cómo funciona Amazon GameLift.

Características principales

- Utilice características de gestión de Amazon GameLift, como el escalado automático, las colas de varias regiones y la ubicación de sesiones de juego.
- Implementación de servidores de juego para ejecutarlos en sistemas operativos de Amazon Linux o Windows Server.
- Administración de sesiones de juego y sesiones de jugador.
- Configuración del seguimiento de estado personalizado para procesos del servidor para detectar problemas y resolver los problemas de rendimiento bajo de los procesos.
- Administre sus recursos de juego mediante las plantillas de AWS CloudFormation para Amazon GameLift.

Alojamiento de Amazon GameLift con Realtime Servers

Utilice Realtime Servers para crear juegos que no necesitan servidores de juegos personalizados. Esta solución de servidor ligera proporciona servidores de juego que puede configurar para adaptarse a su juego.

Para obtener más información sobre el alojamiento de Amazon GameLift con Realtime Servers, consulte Integración de juegos con Servidores en tiempo real de Amazon GameLift.

Características principales

- Utilice características de gestión de Amazon GameLift, como el escalado automático, las colas de varias regiones y la ubicación de sesiones de juego.
- Utilice los recursos de alojamiento de Amazon GameLift y elija el tipo de hardware informático de AWS para sus flotas.
- Aproveche una pila de red completa para la interacción del cliente y servidor de juegos.
- Obtenga la funcionalidad principal del servidor de juegos con lógica de servidor personalizable.
- Realice actualizaciones en directo de la lógica del servidor y las configuraciones de Realtime.

Amazon GameLift FleetIQ para alojamiento en Amazon EC2

Utilice Amazon GameLift FleetIQ para trabajar directamente con sus recursos de alojamiento en Amazon EC2 y Amazon EC2 Auto Scaling. Esto proporciona la ventaja de las optimizaciones de Amazon GameLift para un alojamiento de juegos económico y resistente. Esta solución es para desarrolladores de juegos que necesitan más flexibilidad de la que ofrecen las soluciones totalmente administradas de Amazon GameLift.

Para obtener información sobre cómo Amazon GameLift FleetIQ funciona con Amazon EC2 y EC2 Auto Scaling para el alojamiento de juegos, consulte la Guía para desarrolladores de Amazon GameLift FleetIQ..

Características principales

- Obtenga un equilibrio optimizado de instancias de spot mediante el algoritmo FleetIQ.
- Utilice las características de enrutamiento de jugadores para administrar los recursos de su servidor de juegos de manera eficiente y proporcione una mejor experiencia a los jugadores al unirse a los juegos.
- Escale automáticamente la capacidad de alojamiento en función del uso del jugador.
- Administre directamente instancias de EC2 en su propia Cuenta de AWS.
- Utilice cualquiera de los formatos ejecutables de servidor de juegos compatibles, incluidos Windows, Linux, contenedores y Kubernetes.

Amazon GameLift FlexMatch para emparejamiento

Utilice FlexMatch para crear conjuntos de reglas personalizados para definir los emparejamientos multijugador del juego. FlexMatch utiliza conjuntos de reglas para comparar jugadores compatibles en cada emparejamiento y ofrecer a los jugadores la experiencia multijugador ideal.

Para obtener más información sobre FlexMatch, consulte ¿Qué es Amazon GameLift FlexMatch?

Características principales

- Equilibre la velocidad de creación de emparejamientos con la calidad de emparejamiento.
- Empareje jugadores o equipos según las características definidas.
- Defina reglas para colocar a los jugadores en los emparejamientos en función de la latencia.

Alojamiento de hardware de Amazon GameLift Anywhere

Utilice Amazon GameLift Anywhere para integrar el hardware Anywhere en su entorno en el alojamiento de juegos de Amazon GameLift. Puede integrar flotas de Anywhere y EC2 en las colas de sesión de juego y de emparejador para administrar la ubicación del juego y del emparejamiento en el hardware.

Para obtener más información sobre cómo probar Anywhere, consulte Realización de una prueba de integración con flotas de Amazon GameLift Anywhere. Para obtener más información sobre la configuración de una flota de Anywhere, consulte Configuración de las flotas de Amazon GameLift.

Características principales

- Realice pruebas rápidas e iterativas de las compilaciones de sus servidores y clientes de juegos.
- Utilice el conjunto de herramientas de Amazon GameLift para implementar juegos en su propio hardware.
- Use el hardware que esté más cerca de sus jugadores, en cualquier lugar.

Acceso a Amazon GameLift

Utilice estas herramientas para trabajar con Amazon GameLift.

SDK de Amazon GameLift

Los SDK de Amazon GameLift contienen las bibliotecas necesarias para comunicarse con Amazon GameLift desde los clientes, los servidores y los servicios de juegos. Para obtener más información, consulte Soporte de desarrollo con Amazon GameLift.

SDK del cliente de Realtime de Amazon GameLift

El SDK de cliente de Realtime permite a un cliente de juegos conectarse al servidor de Realtime, unirse a sesiones de juego y mantenerse sincronizado con otros jugadores. Descargue el SDK y obtenga más información sobre cómo realizar llamadas a la API con la API de cliente de Realtime Servers (C#).

Consola de Amazon GameLift

Utilice la AWS Management Console de Amazon GameLift para administrar las implementaciones de juegos, configurar recursos y realizar un seguimiento de las métricas de rendimiento y del uso de los jugadores. La consola de Amazon GameLift ofrece una GUI como alternativa a administrar los recursos mediante programación con la AWS Command Line Interface (AWS CLI).

AWS CLI

Utilice esta herramienta de línea de comandos para realizar llamadas al SDK de AWS, incluida la API de Amazon GameLift. Para obtener más información sobre el uso de AWS CLI, consulte Introducción a la AWS CLI en la Guía del usuario de AWS Command Line Interface.

Precios de Amazon GameLift

Amazon GameLift cobra por las instancias según la duración del uso y por el ancho de banda según la cantidad de datos transferidos. Para obtener una lista completa de los costos y precios de Amazon GameLift, consulte Precios de Amazon GameLift.

Para obtener información sobre cómo calcular el costo del alojamiento de sus juegos o el emparejamiento con Amazon GameLift, consulte Generación de estimaciones de precios de Amazon GameLift, que describe cómo usar AWS Pricing Calculator.

Cómo funciona Amazon GameLift

Este tema trata sobre los componentes principales para el alojamiento de juegos y en él se describe cómo Amazon GameLift pone a disposición de los jugadores los servidores de juegos multijugador.

¿Está listo para preparar su juego para alojarlo en Amazon GameLift? Consulte Hoja de ruta de alojamiento administrado de Amazon GameLift.

Componentes principales

Configurar Amazon GameLift para que aloje un juego implica trabajar con los siguientes componentes: El diagrama de Arquitectura de juegos con Amazon gestionado GameLift muestra las relaciones entre esos componentes.

- Un servidor de juegos es un software del servidor de juegos que se ejecuta en una flota. Cargue la compilación del servidor de juegos o el script en Amazon GameLift e informe de ello a Amazon GameLift. Cuando utilice Amazon GameLift Anywhere o Amazon GameLift FleetIQ, se cargará la compilación del servidor de juegos directamente en el recurso informático.
- Una sesión de juego es un juego en curso con jugadores. Usted debe definir las características básicas de una sesión de juego, tales como su vida útil o el número de jugadores. Después, los jugadores se conectan al servidor de juegos para unirse a una sesión de juego.
- Un cliente de juego es el software del juego que se ejecuta en el dispositivo de un jugador. Un cliente de juegos se conecta a un servidor de juegos a través de servicios de backend para unirse a una sesión de juego en función de la información de conexión que recibe de Amazon GameLift.
- Los servicios de backend son servicios adicionales y personalizados que administran tareas relacionadas con Amazon GameLift. Como práctica recomendada, sus servicios de backend deberían administrar todas las comunicaciones de cliente del juego con Amazon GameLift.

Alojamiento de servidores de juegos

Con Amazon GameLift, puede alojar sus servidores de juegos de tres formas diferentes: Amazon GameLift administrado, Amazon GameLift FleetIQ y Amazon GameLift Anywhere. Para obtener más información sobre Amazon GameLift FleetIQ, consulte ¿Qué es Amazon GameLift FleetIQ?

Puede diseñar una flota que se ajuste a las necesidades de su juego. Para obtener más información sobre el diseño de una flota, consulte Guía de diseño de flotas de Amazon GameLift.

Amazon GameLift administrado

Con Amazon GameLift administrado, puede alojar sus servidores de juegos en los recursos informáticos virtuales de Amazon GameLift denominados instancias. Configure sus recursos de alojamiento. Para ello, cree una flota de instancias e impleméntelas para ejecutar sus servidores de juegos.

Amazon GameLift Anywhere

Con Amazon GameLift Anywhere, podrá alojar sus servidores de juegos en el recurso informático que administra. Configure sus recursos de alojamiento. Para ello, cree una flota de Anywhere que haga referencia al recurso informático.

Alias de flota

Un alias es una designación que puede transferir entre flotas, lo que hace que se pueda disponer de una forma más cómoda de una ubicación de flota genérica. Puede usar un alias para trasladar los clientes de juegos de una flota a otra sin modificar el cliente de juegos. También puede crear un alias de terminal que dirigir al contenido.

Ejecución de sesiones de juego

Después de implementar la compilación del servidor de juegos en una flota y de que Amazon GameLift inicie los procesos del servidor de juegos en cada instancia, la flota podrá alojar sesiones de juego. Amazon GameLift iniciará nuevas sesiones de juego cuando el servicio de cliente de juegos envíe una solicitud de ubicación al servicio de backend o a Amazon GameLift.

Ubicación de las sesiones de juego y algoritmo de FleetIQ

Las colas utilizan el algoritmo de FleetIQ para seleccionar un servidor de juegos disponible en el que alojar una sesión de juego nueva. El componente clave para la ubicación de las sesiones de juegos es la cola de sesiones de juego de Amazon GameLift. Asigne a la cola de sesiones de juego una lista de flotas, que determina dónde puede colocar la cola las sesiones de juego. Para obtener más información sobre las colas de sesiones de juego y cómo diseñarlas para su juego, consulte Diseño de colas de sesiones de juego.

Conexiones de los jugadores a los juegos

Como parte del proceso de ubicación de la sesión de juego, la cola o sesión de juego solicita al servidor de juegos seleccionado que inicie una sesión de juego nueva. El servidor de juegos responde a la solicitud e informa a Amazon GameLift cuando está listo para aceptar conexiones de jugadores. A continuación, Amazon GameLift envía la información de conexión al servicio de backend o al servicio de cliente del juego. Los clientes de juego utilizan esa información para conectarse directamente a la sesión de juego e iniciar el juego.

Escalado de la capacidad de la flota

Una vez que una flota esté activa y lista para alojar sesiones de juego, puede ajustar la capacidad de la flota para hacer frente a la demanda de los jugadores. Le recomendamos que busque un equilibrio entre que todos los jugadores entrantes que buscan un juego rápidamente y gasten de forma excesiva recursos que inactivos.

Amazon GameLift proporciona una herramienta de escalado automático muy eficaz. También es posible configurar manualmente la capacidad de la flota. Para obtener más información, consulte Escalación de la capacidad de alojamiento de Amazon GameLift.

Auto Scaling

Amazon GameLift ofrece dos métodos de escalado automático:

- Escalado automático basado en objetivos
- Escalado automático con políticas basadas en reglas

Características de escalado adicionales

- Protección de la sesión de juego: evita que Amazon GameLift finalice las sesiones de juego que alojan a jugadores activos durante un evento de reducción vertical.
- Límites de escalado: controla el uso general de las instancias mediante el establecimiento de límites mínimos y máximos para el número de instancias en una flota.
- Suspender el escalado automático: suspende el escalado automático en el nivel de ubicación de la flota sin cambiar ni eliminar sus políticas de escalado automático.
- Métricas de escalado: permite realizar un seguimiento del historial de capacidad y los eventos de escalado de una flota.

Supervisión de Amazon GameLift

En cuanto las flotas ya estén operativas, Amazon GameLift recopilará distintos tipos de información que le ayudarán a monitorizar el rendimiento de los servidores de juegos implementados. Puede utilizar esta información para optimizar el uso de los recursos, resolver problemas y obtener información sobre la actividad de los jugadores en los juegos. Amazon GameLift recopila la siguiente información:

- Detalles de la flota, la ubicación, la sesión de juego y la sesión del jugador

- Métricas de uso
- Estado del proceso del servidor
- Registros de sesiones de juego

Para obtener más información sobre la supervisión en Amazon GameLift, consulte Supervisión de Amazon GameLift.

Uso de otros recursos de AWS

Los servidores y las aplicaciones de los juegos pueden comunicarse con otros recursos de AWS. Es posible que utilice un conjunto de servicios web para la autenticación de jugadores o como red social. Para que sus servidores de juegos accedan a recursos de AWS administradas por su Cuenta de AWS, permita explícitamente que Amazon GameLift acceda a sus recursos de AWS.

Amazon GameLift proporciona un par de opciones para la administración de este tipo de acceso. Para obtener más información, consulte Comunicación con otros recursos de AWS de sus flotas.

Cómo se conectan los jugadores a los juegos

Una sesión de juego es una instancia del juego que se ejecuta en Amazon GameLift. Para jugar al juego, un jugador puede buscar y conectarse a una sesión de juego existente o crear una nueva sesión de juego y conectarse a ella. Un jugador se conecta a la sesión mediante la creación de una sesión de jugador para la sesión de juego. Si la sesión de juego está abierta para los jugadores, Amazon GameLift reservará un espacio para el jugador y proporcionará información sobre la conexión. A continuación, el jugador podrá conectarse a la sesión de juego y aprovechar la ranura reservada.

Para obtener información detallada sobre cómo crear y administrar sesiones de juego y de jugador con servidores de juegos personalizados, consulte Añade Amazon GameLift a tu cliente de juegos. Para obtener información sobre cómo conectar jugadores a Realtime Servers, consulte Integración de un cliente de juegos para Realtime Servers.

Amazon GameLift ofrece varias características relacionadas con las sesiones de juego y de jugador.

Alojamiento de sesiones de juego en los mejores recursos disponibles en varias ubicaciones

> Elija entre varias opciones al configurar la manera en que Amazon GameLift selecciona recursos para alojar nuevas sesiones de juego. Si dispone de flotas en varias ubicaciones, puede

diseñar colas de sesiones de juego que coloquen nuevas sesiones de juego en cualquier flota, independientemente de su ubicación.

Control del acceso de jugadores a las sesiones de juego

Configure sesiones de juego para aceptar o rechazar las solicitudes de participaciones de jugadores nuevos, independientemente del número de jugadores conectados.

Uso de datos de jugador y de juego personalizados

Añada datos personalizados a los objetos de la sesión de juego y de jugadores. Al iniciar una nueva sesión de juego, Amazon GameLift transfiere los datos de la sesión de juego a un servidor de juegos. Cuando un jugador se conecta a la sesión del juego, Amazon GameLift transfiere los datos de la sesión del jugador al servidor de juegos.

Filtrado y ordenación de las sesiones de juego disponibles

Utilice la función de búsqueda y ordenación de sesiones para encontrar la mejor coincidencia para un futuro jugador o permita a los jugadores que elijan una opción en la lista de las sesiones de juego disponibles. Utilice la búsqueda y ordenación de sesiones para encontrar las sesiones de juego en función de las características de la sesión. También puede buscar y ordenar en función de sus propios datos de juego personalizados.

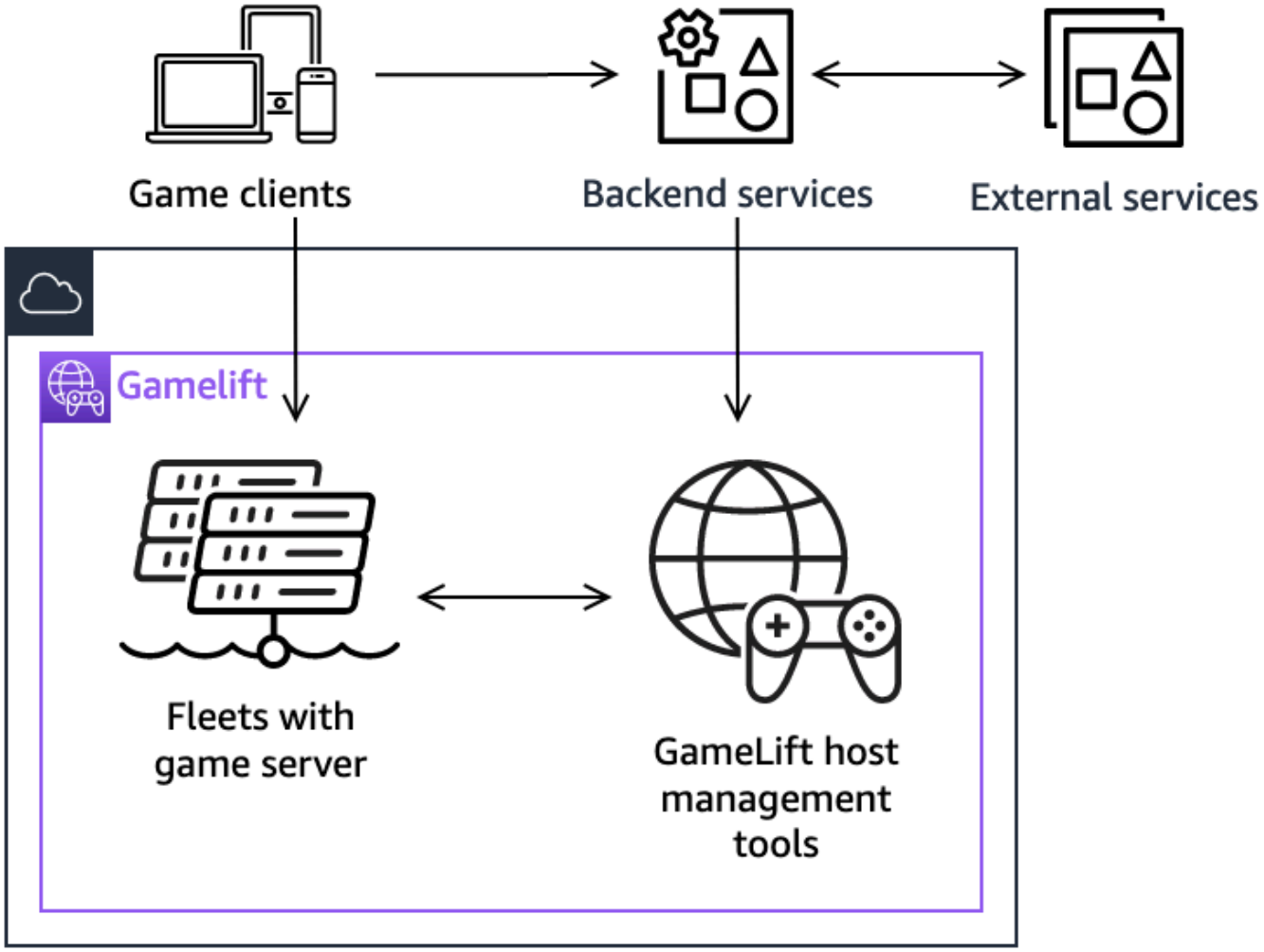
Seguimiento de los datos de uso de juego y de los jugadores

Almacene de forma automática registros de sesiones de juego finalizadas. Cuando realice la integración en Amazon GameLift, podrá configurar el almacenamiento de registros. Para obtener más información, consulte Registro de mensajes del servidor en Amazon GameLift.

Utilice la consola de Amazon GameLift para ver información detallada sobre las sesiones de juego, incluidos los metadatos y la configuración de la sesión, así como los datos de sesiones de jugadores. Para obtener más información, consulte Visualización de datos de sesiones de juego y de jugador y Métricas.

Arquitectura de juegos con Amazon gestionado GameLift

El siguiente diagrama ilustra los componentes clave de una arquitectura de juegos que se aloja mediante la GameLift solución gestionada de Amazon.



Entre los componentes clave de esta arquitectura se incluyen los siguientes:

Clientes de juego

Para unirte a un juego alojado en Amazon GameLift, el cliente del juego debe encontrar primero una sesión de juego disponible. El cliente del juego busca las sesiones de juego existentes, solicita el emparejamiento o inicia una nueva sesión de juego comunicándose con Amazon GameLift a través de un servicio de back-end. El servicio de backend realiza solicitudes a Amazon y GameLift, en respuesta, el servicio recibe información sobre la sesión del juego y la transmite al cliente del juego. A continuación, el cliente de juegos se conecta al servidor de juegos. Para obtener más información, consulte Preparando juegos para Amazon GameLift.

Servicios de backend

Un servicio de backend gestiona la comunicación entre los clientes del juego y Amazon GameLift mediante una llamada a las operaciones de la API del GameLift servicio de Amazon en el AWS SDK. También puede utilizar los servicios de backend para otras tareas específicas del juego, como la autenticación y autorización de jugadores y el control de inventario y divisas. Para obtener más información, consulte Diseño del servicio de cliente de juegos.

Servicios externos

Su juego puede confiar en un servicio externo, por ejemplo, para validar una suscripción. Un servicio externo puede pasar información a los servidores de tus juegos a través de un servicio de back-end y Amazon GameLift.

Servidores de juegos

Subes el software de tu servidor de juegos a Amazon y GameLift, a GameLift continuación, Amazon lo despliega en máquinas de alojamiento para alojar sesiones de juego y aceptar conexiones de jugadores. Los servidores de juegos se comunican con Amazon GameLift para iniciar sesiones de juego, validar a los jugadores recién conectados e informar del estado de las sesiones de juego, las conexiones de los jugadores y los recursos disponibles.

Los servidores de juegos personalizados se comunican con Amazon GameLift mediante el SDK de Amazon GameLift Server. Los clientes del juego se conectan directamente a un servidor de juegos después de recibir los detalles de conexión de Amazon GameLift a través de un servicio de back-end. Para obtener más información, consulte Integración de juegos con servidores de juegos personalizados.

Los servidores de Realtime son servidores de juegos que ejecutan su script personalizado. Al unirse a un juego, un cliente de juegos se conecta directamente a un servidor de Realtime mediante el SDK de cliente de Realtime. Para obtener más información, consulte Integración de juegos con Servidores en tiempo real de Amazon GameLift.

Herramientas de administración de alojamiento

Al configurar y administrar los recursos de alojamiento, los propietarios de juegos utilizan herramientas de administración de alojamiento para gestionar las compilaciones o los scripts de los servidores de juegos, las flotas, los emparejamientos y las colas. El conjunto de GameLift herramientas de Amazon en el AWS SDK y la consola ofrece varias formas de gestionar los recursos de alojamiento. Puede obtener acceso de forma remota a cualquier servidor de juegos individual para la resolución de problemas.

Configuración

Busque ayuda con la configuración de la Cuenta de AWS para utilizar Amazon GameLift con el fin de alojar juegos multijugador.

> **Tip**
>
> Para probar el alojamiento del servidor de juegos de Amazon GameLift, consulte Introducción a Amazon GameLift.

Temas

Configura un Cuenta de AWS

Para empezar a usar Amazon GameLift, crea y configura tu Cuenta de AWS. La creación de una Cuenta de AWS no supondrá ningún gasto. En esta sección se explica cómo crear una cuenta, configurar los usuarios y establecer los permisos.

Temas

Inscríbase en una Cuenta de AWS

Si no tiene uno Cuenta de AWS, complete los siguientes pasos para crearlo.

Para suscribirte a una Cuenta de AWS

1. Abra https://portal.aws.amazon.com/billing/signup.
2. Siga las instrucciones que se le indiquen.

   Parte del procedimiento de registro consiste en recibir una llamada telefónica e indicar un código de verificación en el teclado del teléfono.

   Cuando te registras en una Cuenta de AWS, Usuario raíz de la cuenta de AWSse crea una. El usuario raíz tendrá acceso a todos los Servicios de AWS y recursos de esa cuenta. Como práctica recomendada de seguridad, asigne acceso administrativo a un usuario y utilice únicamente el usuario raíz para realizar tareas que requieren acceso de usuario raíz.

AWS te envía un correo electrónico de confirmación una vez finalizado el proceso de registro. Puede ver la actividad de la cuenta y administrar la cuenta en cualquier momento entrando en https://aws.amazon.com/ y seleccionando Mi cuenta.

Creación de un usuario con acceso administrativo

Después de crear un usuario administrativo Cuenta de AWS, asegúrelo Usuario raíz de la cuenta de AWS AWS IAM Identity Center, habilite y cree un usuario administrativo para no usar el usuario root en las tareas diarias.

Proteja su Usuario raíz de la cuenta de AWS

1. Inicie sesión AWS Management Consolecomo propietario de la cuenta seleccionando el usuario root e introduciendo su dirección de Cuenta de AWS correo electrónico. En la siguiente página, escriba su contraseña.

   Para obtener ayuda para iniciar sesión con el usuario raíz, consulte Signing in as the root user en la Guía del usuario de AWS Sign-In .
2. Active la autenticación multifactor (MFA) para el usuario raíz.

   Para obtener instrucciones, consulte Habilitar un dispositivo MFA virtual para el usuario Cuenta de AWS raíz (consola) en la Guía del usuario de IAM.

Creación de un usuario con acceso administrativo

1. Activar IAM Identity Center.

   Consulte las instrucciones en Activar AWS IAM Identity Center en la Guía del usuario de AWS IAM Identity Center .
2. En IAM Identity Center, conceda acceso administrativo a un usuario.

   Para ver un tutorial sobre su uso Directorio de IAM Identity Center como fuente de identidad, consulte Configurar el acceso de los usuarios con la configuración predeterminada Directorio de IAM Identity Center en la Guía del AWS IAM Identity Center usuario.

Iniciar sesión como usuario con acceso de administrador

- Para iniciar sesión con el usuario de IAM Identity Center, utilice la URL de inicio de sesión que se envió a la dirección de correo electrónico cuando creó el usuario de IAM Identity Center.

  Para obtener ayuda para iniciar sesión con un usuario del Centro de identidades de IAM, consulte Iniciar sesión en el portal de AWS acceso en la Guía del AWS Sign-In usuario.

Concesión de acceso a usuarios adicionales

1. En IAM Identity Center, cree un conjunto de permisos que siga la práctica recomendada de aplicar permisos de privilegios mínimos.

   Para conocer las instrucciones, consulte Create a permission set en la Guía del usuario de AWS IAM Identity Center .
2. Asigne usuarios a un grupo y, a continuación, asigne el acceso de inicio de sesión único al grupo.

   Para conocer las instrucciones, consulte Add groups en la Guía del usuario de AWS IAM Identity Center .

Administrar los permisos de usuario para Amazon GameLift

Crea usuarios adicionales o amplía los permisos de acceso a los usuarios existentes según sea necesario para tus GameLift recursos de Amazon. Como práctica recomendada (Prácticas de seguridad en IAM), aplique permisos con privilegios mínimos a todos los usuarios. Para obtener

información sobre la sintaxis de los permisos, consulte Ejemplos de permisos de IAM para Amazon GameLift.

Sigue las siguientes instrucciones para configurar los permisos de usuario en función de la forma en que gestiones los usuarios de tu AWS cuenta.

Para dar acceso, agregue permisos a los usuarios, grupos o roles:

- Usuarios y grupos en AWS IAM Identity Center:

  Cree un conjunto de permisos. Siga las instrucciones de Creación de un conjunto de permisos en la Guía del usuario de AWS IAM Identity Center .
- Usuarios administrados en IAM a través de un proveedor de identidades:

  Cree un rol para la federación de identidades. Siga las instrucciones descritas en Creación de un rol para un proveedor de identidad de terceros (federación) en la Guía del usuario de IAM.
- Usuarios de IAM:
  - Cree un rol que el usuario pueda aceptar. Siga las instrucciones descritas en Creación de un rol para un usuario de IAM en la Guía del usuario de IAM.
  - (No recomendado) Adjunte una política directamente a un usuario o añada un usuario a un grupo de usuarios. Siga las instrucciones descritas en Adición de permisos a un usuario (consola) de la Guía del usuario de IAM.

Cuando trabaje con usuarios de IAM, una práctica recomendada es adjuntar siempre los permisos a roles o grupos de usuarios, no a usuarios individuales.

Para configurar el acceso mediante programación para usuarios, realice el siguiente procedimiento:

Los usuarios necesitan acceso programático si quieren interactuar con personas AWS ajenas a. AWS Management Console La forma de conceder el acceso programático depende del tipo de usuario que acceda. AWS

Para conceder acceso programático a los usuarios, elija una de las siguientes opciones.

¿Qué usuario necesita acceso programático?	Para	Mediante
Identidad del personal (Usuarios administrados en el IAM Identity Center)	Usa credenciales temporales para firmar las solicitudes programáticas a los AWS CLI AWS SDK o las API. AWS	Siga las instrucciones de la interfaz que desea utilizar: • Para ello AWS CLI, consulte Configuración del uso AWS IAM Identity Center en AWS CLI la Guía del AWS Command Line Interface usuario. • Para obtener AWS información sobre los SDK, las herramientas y AWS las API, consulte la autenticación del IAM Identity Center en la Guía de referencia de AWS los SDK y las herramientas.
IAM	Utilice credenciales temporales para firmar las solicitudes programáticas a los AWS SDK o las AWS CLI API. AWS	Siga las instrucciones de Uso de credenciales temporales con AWS recursos de la Guía del usuario de IAM.
IAM	(No recomendado) Utilice credenciales de larga duración para firmar las solicitudes programáticas a los AWS CLI AWS SDK o las API. AWS	Siga las instrucciones de la interfaz que desea utilizar: • Para ello AWS CLI, consulte Autenticación con credenciales de usuario de IAM en la Guía del usuario.AWS Command Line Interface • Para obtener información AWS sobre los SDK y las herramientas, consulte

¿Qué usuario necesita acceso programático?	Para	Mediante
		Autenticarse con credencia les de larga duración en la Guía de referencia de los AWS SDK y las herramien tas. • Para obtener informaci ón AWS sobre las API, consulte Administrar las claves de acceso para los usuarios de IAM en la Guía del usuario de IAM.

Si utilizas claves de acceso, consulta las prácticas recomendadas para gestionar las claves de AWS acceso.

Configuración de acceso mediante programación para su juego

La mayoría de los juegos utilizan servicios de backend para comunicarse con Amazon GameLift mediante los AWS SDK. Por ejemplo, utilice un servicio de backend (que actúe en nombre de los clientes de juegos) para solicitar sesiones de juego, ubicar a los jugadores en los juegos y realizar otras tareas. Estos servicios necesitan acceso programático y credenciales de seguridad para autenticar las llamadas a las API de GameLift servicio de Amazon.

En el caso de Amazon GameLift, este acceso se gestiona mediante la creación de un usuario reproductor en AWS Identity and Access Management (IAM). Administre los permisos de usuario del jugador mediante una de las siguientes opciones:

- Cree un rol de IAM con permisos de usuario del jugador y permita que el usuario jugador asuma el rol cuando sea necesario. El servicio de backend debe incluir código para asumir esta función antes de realizar solicitudes a Amazon GameLift. De acuerdo con las prácticas recomendadas de seguridad, los roles proporcionan un acceso limitado y temporal. Puede usar roles para cargas de trabajo que se ejecuten en AWS recursos (roles de IAM) o fuera de ellos AWS (roles de IAM en cualquier lugar).

- Cree un grupo de usuarios de IAM con permisos de usuario jugador y añada su usuario jugador al grupo. Esta opción proporciona a tu jugador credenciales de usuario a largo plazo, que el servicio de backend debe almacenar y utilizar al comunicarse con Amazon GameLift.

Para ver la sintaxis de la política de permisos, consulte Ejemplos de permisos de usuario de un jugador.

Para obtener más información sobre la administración de los permisos para su uso en una carga de trabajo, consulte Identidades de IAM: credenciales temporales en IAM.

Ejemplos de permisos de IAM para Amazon GameLift

Utilice la sintaxis de estos ejemplos para configurar los permisos de AWS Identity and Access Management (IAM) para los usuarios que necesitan acceder a los recursos de Amazon GameLift. Para obtener más información sobre la administración de permisos del usuario, consulte Administrar los permisos de usuario para Amazon GameLift. Cuando administre los permisos de los usuarios fuera del Centro de identidades de IAM, se recomienda adjuntar siempre los permisos a los roles o grupos de usuarios de IAM, no a los usuarios individuales.

Si utiliza Amazon GameLift FleetIQ como solución independiente, consulte Configuración de la Cuenta de AWS para Amazon GameLift FleetIQ.

Ejemplos de permisos de administrador

Estos ejemplos proporcionan al usuario acceso completo para administrar los recursos de alojamiento de juegos de Amazon GameLift.

Example Sintaxis de los permisos de recursos de Amazon GameLift

El siguiente ejemplo amplía el acceso a todos los recursos de Amazon GameLift.

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

Example Sintaxis de los permisos de recursos de Amazon GameLift con soporte para regiones que no están habilitadas de forma predeterminada

El siguiente ejemplo amplía el acceso a todos los recursos y regiones de AWS de Amazon GameLift que no están habilitados de forma predeterminada. Para obtener más información sobre las regiones que no están habilitadas de forma predeterminada y cómo habilitarlas, consulte Administración de Regiones de AWS en la Referencia general de AWS.

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

Example Sintaxis de los permisos **`PassRole`** y recursos de Amazon GameLift

El siguiente ejemplo amplía el acceso a todos los recursos de Amazon GameLift y permite a un usuario transferir un rol de servicio de IAM a Amazon GameLift. Un rol de servicio otorga a Amazon GameLift una capacidad limitada para acceder a otros recursos y servicios en su nombre, tal y como se describe en Configurar un rol de servicio de IAM para Amazon GameLift. Por ejemplo, al responder a una solicitud de `CreateBuild`, Amazon GameLift necesita acceder a sus archivos de compilación en un bucket de Amazon S3. Para obtener más información sobre la acción `PassRole`, consulte IAM: pasar un rol de IAM a un servicio de AWS específico en la Guía del usuario de IAM.

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

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

Ejemplos de permisos de usuario de un jugador

Estos ejemplos permiten a un servicio de backend u otra entidad realizar llamadas a la API de Amazon GameLift. Incluyen los escenarios más comunes para administrar las sesiones de juego, las sesiones de los jugadores y el emparejamiento. Para obtener más información, consulte Configuración de acceso mediante programación para su juego.

Example Sintaxis de los permisos de colocación de las sesiones de juego

El siguiente ejemplo amplía el acceso a las API de Amazon GameLift que utilizan colas de ubicación de sesiones de juego para crear sesiones de juego y administrar las sesiones de los jugadores.

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

Example Sintaxis de los permisos de emparejamiento

El siguiente ejemplo amplía el acceso a las API de Amazon GameLift que administran las actividades de emparejamiento de FlexMatch. FlexMatch empareja a los jugadores para sesiones de juego nuevas o existentes e inicia la ubicación de las sesiones de juego para los juegos alojados en Amazon GameLift. Para obtener más información sobre FlexMatch, consulte ¿Qué es 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 Sintaxis de los permisos de ubicación manual de las sesiones de juego

El siguiente ejemplo amplía el acceso a las API de Amazon GameLift que crean manualmente sesiones de juego y sesiones de los jugadores en flotas especificadas. Este escenario admite los juegos que no utilizan colas de ubicación, como los juegos en los que los jugadores pueden unirse eligiendo una opción de entre una lista de sesiones de juego disponibles (el método de «lista y selección»).

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

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

Configurar un rol de servicio de IAM para Amazon GameLift

Algunas GameLift funciones de Amazon requieren que extiendas el acceso limitado a AWS los recursos de tu propiedad. Para ello, puede crear un rol de AWS Identity and Access Management (IAM). Un rol de IAM es una identidad de IAM que puede crear en su cuenta y que tiene permisos específicos. Un rol de IAM es similar a un usuario de IAM en que se trata de una identidad de AWS con políticas de permisos que determinan lo que la identidad puede hacer y lo que no en AWS. No obstante, en lugar de asociarse exclusivamente a una persona, la intención es que cualquier usuario pueda asumir un rol que necesite. Además, un rol no tiene asociadas credenciales a largo plazo estándar, como una contraseña o claves de acceso. En su lugar, cuando se asume un rol, este proporciona credenciales de seguridad temporales para la sesión de rol.

En este tema se explica cómo crear un rol que puedas usar con tus flotas GameLift gestionadas por Amazon. Si utiliza Amazon GameLift FleetIQ para optimizar el alojamiento de juegos en sus instancias de Amazon Elastic Compute Cloud (Amazon EC2), consulte Configurar su versión para Amazon FleetIQ. Cuenta de AWS GameLift

En el siguiente procedimiento, cree un rol con una política de permisos personalizada y una política de confianza que permita GameLift a Amazon asumir el rol.

Creación de un rol de IAM personalizado

Paso 1: Creación de una política de permisos.

Para utilizar el editor de política de JSON para crear una política

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la izquierda, seleccione Políticas.

   Si es la primera vez que elige Políticas, aparecerá la página Bienvenido a políticas administradas. Elija Comenzar.

3. En la parte superior de la página, seleccione Crear política.
4. En la sección Editor de políticas, seleccione la opción JSON.
5. Introduzca o pegue un documento de política de JSON. Para obtener más información sobre el lenguaje de la política de IAM, consulte Referencia de políticas JSON de IAM.
6. Resuelva las advertencias de seguridad, errores o advertencias generales generadas durante la validación de política y luego elija Siguiente.

> **Note**
>
> Puede alternar entre las opciones Visual y JSON del editor en todo momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política, con el fin de optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de política en la Guía del usuario de IAM.

7. (Opcional) Al crear o editar una política en la AWS Management Console, se puede generar una plantilla de política JSON o YAML que se puede utilizar en plantillas de AWS CloudFormation.

   Para ello, en el editor de políticas, selecciona Acciones y, a continuación, selecciona Generar CloudFormation plantilla. Para obtener más información sobre AWS CloudFormation, consulte la Referencia de tipos de recursos de AWS Identity and Access Management en la Guía del usuario de AWS CloudFormation.
8. Cuando haya terminado de agregar permisos a la política, seleccione Siguiente.
9. En la página Revisar y crear, introduzca el Nombre de la política y la Descripción (opcional) para la política que está creando. Revise los Permisos definidos en esta política para ver los permisos que concede la política.
10. (Opcional) Agregar metadatos a la política al adjuntar las etiquetas como pares de clave-valor. Para obtener más información sobre el uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM en la Guía de usuario de IAM.
11. Elija Create Policy (Crear política) para guardar la nueva política.

Paso 2: Crea un rol que Amazon GameLift pueda asumir.

1. En el panel de navegación de la consola de IAM, seleccione Roles y, a continuación, elija Crear rol.

2. En la página Seleccionar entidad de confianza, elija la opción Política de confianza personalizada. Con esta selección se abre el editor de Política de confianza personalizada.
3. Sustituya la sintaxis JSON predeterminada por la siguiente y, a continuación, elija Siguiente para continuar.

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

4. En la página Añadir permisos, busque y seleccione la política de permisos creada en el paso 1. Elija Siguiente para continuar.
5. En la página Nombrar, revisar y crear, introduzca el Nombre de rol y una Descripción (opcional) para el rol que está creando. Revise las Entidades de confianza y los Permisos agregados.
6. Seleccione Crear rol para guardar el nuevo rol.

Sintaxis de la política de permisos

- Permisos para GameLift que Amazon asuma la función de servicio

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

- Permisos para acceder a las regiones de AWS que no están habilitadas de forma predeterminada

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

Soporte de desarrollo con Amazon GameLift

Amazon GameLift proporciona un conjunto de SDK que puedes usar con tus soluciones de alojamiento de juegos gestionado. Usa GameLift los SDK de Amazon para añadir la funcionalidad necesaria a los servidores de juegos multijugador, los clientes de juegos y los servicios de juegos que necesitan interactuar con el servicio de GameLift alojamiento de Amazon.

Para obtener la información más reciente sobre las versiones de Amazon GameLift SDK y la compatibilidad de los SDK, consulteNotas de GameLift lanzamiento de Amazon.

Para servidores de juegos personalizados, realice el siguiente procedimiento:

Crea e implementa servidores de juegos personalizados de 64 bits con el SDK para GameLift servidores de Amazon. Los servidores de juegos integrados con el SDK del servidor y desplegados para el alojamiento pueden comunicarse con el GameLift servicio de Amazon para iniciar y gestionar las sesiones de juego. Para obtener información sobre la integración del SDK del servidor, consulte los temas de Preparando juegos para Amazon GameLift.

Sistemas operativos de desarrollo

- Windows
- Linux

Lenguajes de programación admitidos

Amazon GameLift proporciona el SDK de servidor para los siguientes idiomas. Descarga los SDK de servidor. Para obtener información detallada y específica de cada versión, consulte los archivos Léame incluidos en cada paquete.

- SDK del servidor C++
  - Referencia del SDK
  - Integración del SDK
- SDK del servidor C# (las versiones pueden ser compatibles con .NET 4 y .NET 6)
  - Referencia del SDK
  - Integración del SDK
- Go
  - Referencia del SDK
  - Integración del SDK

Motores de juegos admitidos

Utilice los SDK específicos del idioma con cualquier motor que admita bibliotecas de C++, C# o Go. Además, Amazon GameLift proporciona los siguientes complementos para motores de juegos: Descargar GameLift complementos de Amazon

- Unity
  - El complemento del SDK del servidor C# para Unity es un complemento ligero con bibliotecas prediseñadas que puede instalar mediante el administrador de paquetes de Unity. Utilice este complemento con las siguientes versiones de Unity: 2020.3 LTS, 2021.3 LTS y 2022.3 LTS para Windows y Mac OS. Es compatible con los perfiles .NET Framework y .NET Standard de Unity, con .NET Standard 2.1 y .NET 4.x.
    - Integración de Amazon GameLift en un proyecto de Unity

  - El complemento independiente para Unity 2021.3 LTS y 2022.3 LTS es un complemento con todas las funciones con las bibliotecas del SDK de C# creadas para Unity y elementos de la GUI para configurar e implementar los recursos de Amazon para el alojamiento. GameLift
    - Guía GameLift del complemento de Amazon para Unity para el SDK de servidor 5.x
    - Referencia del SDK del servidor de Amazon GameLift para C#
- Unreal Engine
  - El complemento del SDK del servidor C++ para Unreal es un complemento ligero que consta del código fuente de Unreal para C++ y que se puede compilar en bibliotecas para su uso con las versiones 4, 5 y 5.1 de Unreal Engine.
    - Integre Amazon GameLift en un proyecto de Unreal Engine
    - Referencia del SDK del servidor de Unreal Engine de Amazon GameLift 5.x
  - El complemento independiente para Unreal Engine 5.0, 5.1 y 5.2 es un complemento con todas las funciones que incluye las bibliotecas del SDK y el SDK del servidor C++ para Unreal. AWS El complemento se instala en el editor Unreal, con elementos de interfaz de usuario y materiales de apoyo para configurar e implementar GameLift los recursos de Amazon para el alojamiento.
    - Integración de juegos con el GameLift complemento de Amazon para Unreal Engine
    - Referencia del SDK del servidor de Unreal Engine de Amazon GameLift 5.x

Sistemas operativos de servidor de juegos

Usa el SDK de Amazon GameLift Server para crear servidores de juegos que se ejecuten en las siguientes plataformas:

- Windows Server 2016
- Amazon Linux 2023
- Amazon Linux 2 (AL2)
- Windows Server 2012 (consulte las GameLift Preguntas frecuentes de Amazon para Windows 2012)
- Amazon Linux (AL1) (consulte las GameLift Preguntas frecuentes de Amazon para AL1)

Para servicios de cliente personalizados, realice el siguiente procedimiento:

Cree servicios de cliente personalizados de 64 bits mediante el AWS SDK con la GameLift API de Amazon. Este SDK permite a los servicios de cliente gestionar las sesiones de juego y unir a los

jugadores a los juegos alojados en Amazon GameLift. Para empezar, descarga el AWS SDK. Para obtener más información sobre el uso del SDK con Amazon GameLift, consulta la referencia de la GameLift API de Amazon.

Para servidores de Realtime, realice el siguiente procedimiento:

Configure e implementa servidores de Realtime para alojar los juegos multijugador. Para permitir que sus clientes de juegos se conecten a servidores de Realtime, utilice el SDK de Amazon GameLift Realtime Client. Los clientes de juego utilizan este SDK para intercambiar mensajes con un servidor de Realtime y con otros clientes de juego que se conectan al servidor. Para empezar, descargue el SDK de Amazon GameLift Realtime Client. Para obtener información sobre la configuración, consulte Integración de un cliente de juegos para Realtime Servers.

Compatibilidad con SDK

El SDK de cliente de Realtime contiene código fuente para los siguientes lenguajes:

- C# (.NET)

Entornos de desarrollo

Compile el SDK a partir del código adecuado para los siguientes sistemas operativos de desarrollo y motores de videojuegos compatibles:

- Sistemas operativos: Windows, Linux, Android e iOS
- Motores de juegos: Unity y motores que admiten bibliotecas de C#

Sistemas operativos de servidor de juegos

Puede implementar servidores de Realtime en los recursos de alojamiento que se ejecuten en las siguientes plataformas:

- Amazon Linux
- Amazon Linux 2

Gestión de los costos de alojamiento de juegos

Su factura de AWS refleja los costos de alojamiento de sus juegos. Puede ver los cargos estimados del mes actual y los cargos finales de los meses anteriores en la consola de facturación en https://

console.aws.amazon.com/billing/. Para obtener más información sobre las herramientas y recursos que les ayuden a administrar los costos de AWS, consulte la Guía del usuario de AWS Billing. Esta guía le ayudará a revisar el consumo de recursos, a establecer el uso en un futuro y a determinar sus necesidades de escalado.

En concreto, ten en cuenta estos consejos para ayudarte a gestionar el coste de los GameLift servicios de Amazon.

Creación de alertas de facturación para supervisar el uso

Configura una alerta de uso de la capa AWS gratuita para que te notifique cuando tu uso se acerque o supere los límites de la capa gratuita para Amazon GameLift y otrosServicios de AWS. Puede configurar las alertas para que tomen medidas en función de sus niveles de uso. Por ejemplo, puedes establecer automáticamente el presupuesto en cero cuando alcance el límite de nivel gratuito.

También puedes configurar alertas de CloudWatch facturación de Amazon para recibir notificaciones cuando el uso alcance los umbrales personalizados.

Para obtener más información, consulte estos temas en la Guía del usuario de AWS Billing:

- Seguimiento del uso de nivel gratuito de AWS
- Preferencias de alertas de facturación

Realiza un seguimiento de los costes por GameLift flota de Amazon

Utilice etiquetas de asignación de AWS costes para organizar y realizar un seguimiento de los costes de alojamiento de juegos en función de las flotas de GameLift Amazon EC2 y otros recursos de EC2. Al etiquetar las flotas, ya sea de forma individual o por grupos, puede crear informes de asignación de costos que clasifiquen los costos en función de la etiqueta asignada. Puede utilizar este tipo de informe para identificar cómo las flotas contribuyen a los costos de alojamiento. También puede usar etiquetas para filtrar vistas en AWS Cost Explorer.

Para obtener más información, consulte estos temas:

- Uso de etiquetas de asignación de costos de AWS en la Guía del usuario de AWS Billing
- Análisis de los costos con AWS Cost Explorer, Guía del usuario de AWS Cost Management

Establecimiento de la capacidad de flota no utilizada en cero

Las flotas pueden seguir incurriendo en costos incluso cuando no se utilizan para alojar sesiones de juego. Para evitar gastos innecesarios, le recomendamos reducir verticalmente su flota a cero cuando no la utilice. Si utiliza el escalado automático, suspenda esta actividad y configure manualmente la capacidad de la flota.

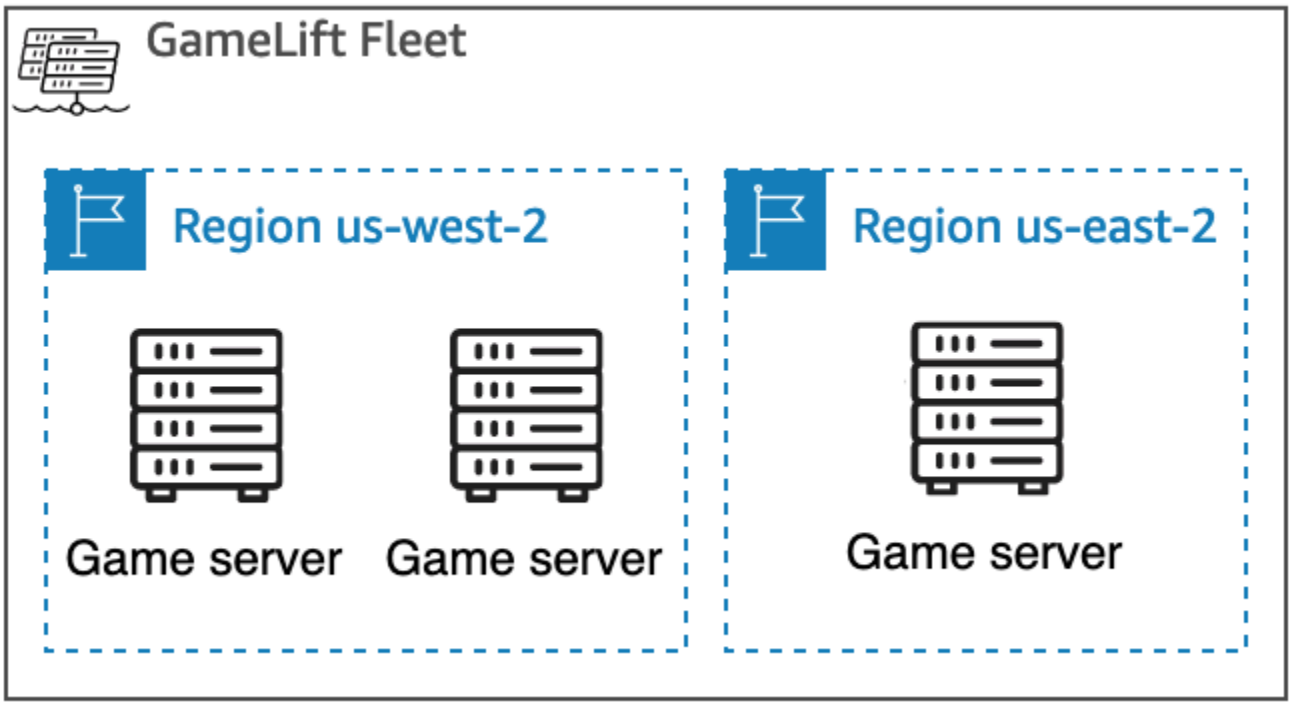
Ubicaciones GameLift de alojamiento de Amazon

Amazon GameLift está disponible en varias Zonas Regiones de AWS y en Zonas Locales. Para obtener una lista completa de las ubicaciones, consulta los GameLiftpuntos de conexión y las cuotas de Amazon en. Referencia general de AWS

GameLift Alojamiento en Amazon

Cuando creas una GameLift flota de Amazon, Amazon GameLift crea los recursos de la flota en tu flota actualRegión de AWS. Amazon GameLift llama a esta región la región de origen de la flota. Para administrar una flota, acceda a ella desde su región de origen.

Las flotas con varias ubicaciones permiten implementar instancias en otras ubicaciones además de en la región de origen de la flota. Con las flotas con múltiples ubicaciones, puedes gestionar la capacidad de cada ubicación de forma individual y organizar las sesiones de juego por ubicación. Las flotas con múltiples ubicaciones pueden tener ubicaciones remotas en cualquier región o zona local que Amazon GameLift admita. El siguiente diagrama muestra una flota con varias ubicaciones con recursos en dos regiones. En el diagrama, la región `us-west-2` incluye dos servidores de juegos, y la región `us-east-2` un servidor de juegos.



Si opta por utilizar una flota con varias ubicaciones con instancias en regiones que no estén habilitadas de forma predeterminada, deberá habilitar esas regiones en su Cuenta de AWS. Además, tu política de GameLift administrador de Amazon debe permitir esta `ec2:DescribeRegions` acción. Para obtener más información sobre las regiones que no están habilitadas de forma predeterminada y cómo habilitarlas, consulte Administración de Regiones de AWS en la Referencia general de AWS. Para ver un ejemplo de política con regiones que no están habilitadas de forma predeterminada, consulte Ejemplos de permisos de administrador.

> **Important**
>
> Para utilizar las regiones que no están habilitadas de forma predeterminada, habilítelas en su Cuenta de AWS.
>
> - Las flotas con regiones que no están habilitadas, y que haya creado antes del 28 de febrero de 2022, no se verán afectadas.
> - Para crear nuevas flotas con varias ubicaciones o actualizar las existentes, habilite primero las regiones que desee utilizar.

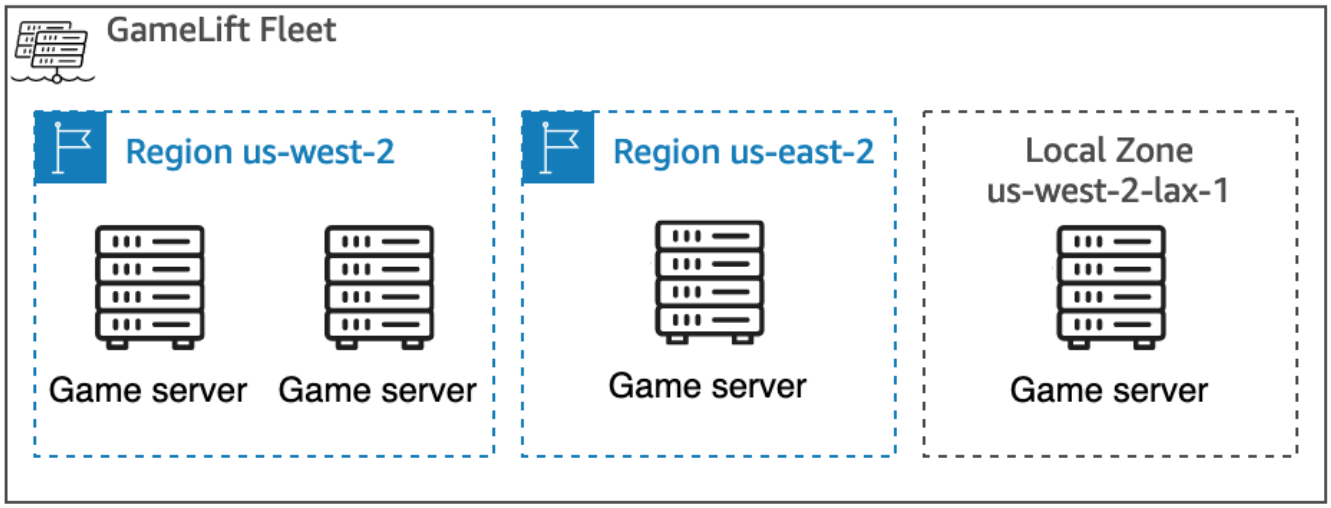
Para ubicar las sesiones de juego, puedes crear colas de sesiones de juego en cualquier ubicación que Amazon GameLift admita. Amazon GameLift coloca las sesiones de juego desde la ubicación en la que creaste la cola.

Zonas locales

Una zona local es una extensión de una Región de AWS que se encuentra geográficamente cerca de los usuarios. Las zonas locales tienen sus propias conexiones a Internet y también son compatibles con AWS Direct Connect, por lo que los recursos creados en una zona local pueden ofrecer a los usuarios locales comunicaciones de baja latencia. Para obtener más información, consulte AWS Local Zones.

El código de una zona local es el código de su región seguido de un identificador que indica su ubicación física. Por ejemplo, la zona local `us-west-2-lax-1` está en Los Ángeles. Para ver una lista de las zonas locales disponibles, consulte Local Zones disponibles.

Amazon GameLift aloja tus juegos en cada una de las ubicaciones que elijas para tu flota. El siguiente diagrama muestra una flota con dos servidores de juegos en la región `us-west-2`, uno en la región `us-east-2` y otro en la zona local `us-west-2-lax-1`.



Local Zones disponibles

En la tabla siguiente se describen las zonas locales disponibles y sus ubicaciones físicas.

Zona local	Ubicación (área metropolitana)	
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 Ángeles	
us-west-2-phx-1	Phoenix	

Amazon GameLift Anywhere

Puedes usar Amazon GameLift Anywhere para crear flotas con tu propio hardware y administrar tus compilaciones de juegos, scripts, servidores de juegos y clientes con Amazon GameLift. Amazon GameLift Anywhere está disponible en todas las regiones que Amazon GameLift admite. Para obtener más información sobre cómo crear una flota de Anywhere y probar la integración del servidor de juegos, consulte Crea una GameLift Anywhere flota de Amazon y Realización de una prueba de integración con flotas de Amazon GameLift Anywhere.

Con Amazon GameLift Anywhere, crea ubicaciones personalizadas que representan la ubicación física del hardware que utiliza para alojar los servidores de juegos GameLift integrados de Amazon.

Amazon GameLift FlexMatch

Pues FlexMatch, puedes organizar sesiones de juego generadas por partidos en cualquier ubicación compatible con Amazon GameLift. La actividad de emparejamiento real se lleva a cabo en el lugar Región de AWS donde elijas crear tus recursos de emparejamiento. Amazon GameLift envía las solicitudes de coincidencia al emparejador y las procesa en esa ubicación. Para obtener más información sobre Amazon GameLift FlexMatch, consulta ¿Qué es Amazon GameLift FlexMatch?

Regiones de AWSque respalden FlexMatch los recursos

Amazon GameLift en China

Si utilizas Amazon GameLift para recursos en la región de China (Pekín), gestionada por Sinnet, o en la región de China (Ningxia), gestionada por NWCD, debes tener una cuenta independiente AWS (China). Tenga en cuenta que algunas características no están disponibles en las regiones de China. Para obtener más información sobre el uso de Amazon GameLift en estas regiones, consulta los siguientes recursos:

- Amazon Web Services en China
- Amazon GameLift (Introducción a Amazon Web Services en China)

Introducción a Amazon GameLift

Le recomendamos que pruebe los siguientes ejemplos antes de utilizar Amazon GameLift para su propio juego. El ejemplo del servidor de juegos personalizado le proporciona experiencia con el alojamiento de juegos en la consola de Amazon GameLift. El ejemplo de Realtime Servers muestra cómo preparar un juego para su alojamiento con Realtime Servers.

Para empezar a utilizar Amazon GameLift para su propio juego, consulte Hoja de ruta de alojamiento administrado de Amazon GameLift.

Ejemplo de servidor de juegos personalizado

En este ejemplo, se muestra un juego personalizado en directo en Amazon GameLift. En el ejemplo se le guiará por los siguientes pasos:

- Creación de una compilación de juego de ejemplo
- Creación de una flota para ejecutar el servidor de juegos
- Conexión al servidor de juegos a partir del cliente de juego de ejemplo
- Revisión de las métricas de la flota y las sesiones de juego

Después de esos pasos, puede iniciar varios clientes de juego y reproducirlos para generar datos de alojamiento. A continuación, puede explorar la consola de Amazon GameLift para ver los recursos de alojamiento, realizar un seguimiento de las métricas y experimentar con distintas formas de escalar la capacidad de alojamiento.

Para empezar, inicie sesión en la consola de Amazon GameLift.

Juego de ejemplo de Realtime Servers

Este ejemplo es un juego multijugador completo denominado «Mega Frog Race», con el código fuente incluido. El ejemplo muestra cómo integrar el cliente de juego con Realtime Servers. También puede usar este juego de ejemplo como punto de partida para experimentar con otras características de Amazon GameLift, como FlexMatch.

Para ver el tutorial práctico, consulte Creación de servidores para juegos móviles multijugador con solo unas líneas de JavaScript del blog de AWS para juegos.

Para ver el código fuente de Mega Frog Race, consulte el repositorio de GitHub.

En el código fuente se incluyen las partes siguientes:

- Cliente de juego: código fuente para el cliente de juego C++, creado en Unity. El cliente de juego obtiene la información de conexión de la sesión de juego, se conecta al servidor e intercambia actualizaciones con otros jugadores.
- Servicio de backend: código fuente de una función AWS Lambda que administra las llamadas directas a la API a Amazon GameLift.
- Script en tiempo real: script de origen que configura una flota de Realtime Servers para el juego. Este script incluye la configuración mínima necesaria para que los Realtime Servers se comuniquen con Amazon GameLift y alojen juegos.

Hoja de ruta de alojamiento administrado de Amazon GameLift

Este tema le servirá de ayuda para elegir entre las diferentes opciones de alojamiento de Amazon GameLift para su juego multijugador basado en la sesión. En el resto de los temas de esta sección se explicará cómo utilizar Amazon GameLift para el alojamiento administrado.

Antes de comenzar a preparar el lanzamiento del juego para la producción, complete el cuestionario de lanzamiento para empezar a trabajar con el equipo de Amazon GameLift.

Temas

Elección de una opción de alojamiento

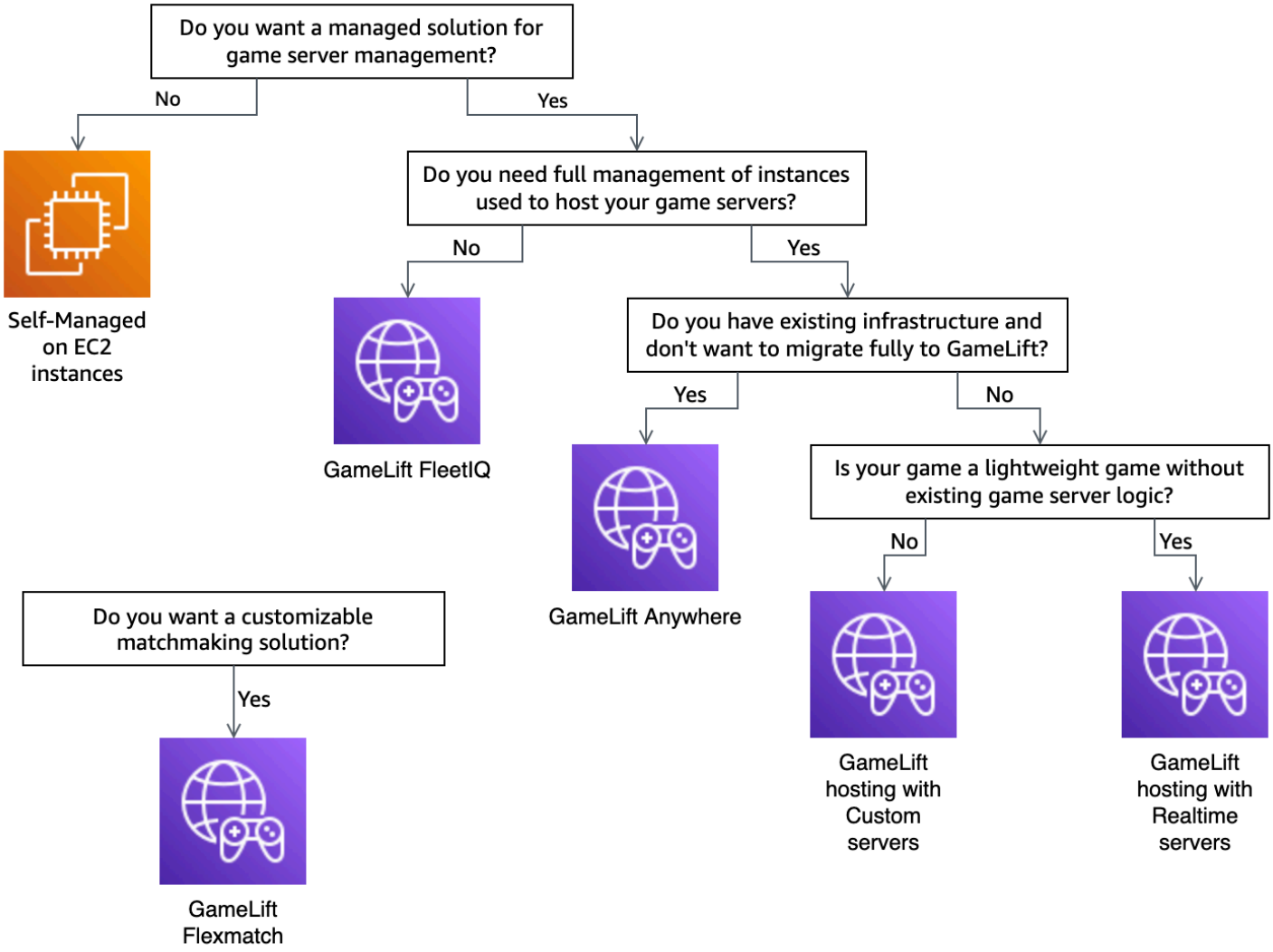
En el siguiente diagrama de flujo se formulan preguntas que le llevarán a la solución más adecuada de Amazon GameLift para su caso de uso.

1. ¿Desea una solución administrada para la gestión de servidores de juegos?
   - Sí. Continúe con el segundo paso.
   - No. Considere la posibilidad de utilizar servidores de juegos autoadministrados en las instancias de Amazon EC2.
2. ¿Necesita tener el control total de las instancias que alojan sus servidores de juegos?
   - Sí. Considere Amazon GameLift FleetIQ.
   - No. Continúe con el paso 3.
3. ¿Dispone de una infraestructura existente que desea utilizar con Amazon GameLift?

- Sí. Considere Amazon GameLift Anywhere.
- No. Continúe con el paso 4.

4. ¿Su juego tiene un formato ligero y no sigue la lógica del servidor de juegos existente?

   - Sí. Considere los servidores de Realtime.
   - No. Considere la posibilidad de usar servidores personalizados.



Aquí dispone de más información sobre algunas de las opciones de alojamiento de Amazon GameLift mencionadas en el diagrama de flujo:

Amazon GameLift Anywhere

Utilice Amazon GameLift Anywhere para alojar juegos en su propio hardware con las ventajas de las herramientas de administración de Amazon GameLift. También puede usar flotas de Anywhere para probar los servidores de juegos de forma iterativa. Para obtener más información, consulte Crea una GameLift Anywhere flota de Amazon.

Amazon GameLift administrado

Existen dos opciones para el alojamiento administrado de Amazon GameLift:

Servidores personalizados: Amazon GameLift aloja el servidor personalizado que ejecuta el archivo binario de su servidor de juegos.

Realtime Servers: Amazon GameLift aloja el servidor de juegos de formato ligero.

Amazon GameLift FleetIQ

En el diagrama de flujo, una migración mediante Lift and shift hace referencia a una migración en la que no se pueden realizar cambios en la arquitectura del juego. El uso de Amazon GameLift FleetIQ requiere menos cambios en la implementación existente y proporciona herramientas de Amazon GameLift para la administración de flotas. Para obtener más información, consulte la Guía para desarrolladores de Amazon GameLift FleetIQ.

Si decide utilizar Amazon GameLift Anywhere o Amazon GameLift administrado, diríjase a Preparación del juego para Amazon GameLift.

Preparación del juego para Amazon GameLift

En este tema se describen los pasos para preparar el juego multijugador para su integración con el alojamiento administrado de Amazon GameLift. Para preparar el juego, debe activar la comunicación entre este y Amazon GameLift.

Preparación del servidor de juegos personalizado

Para iniciar y detener las sesiones de juego, y para realizar otras tareas, un servidor de juegos debe ser capaz de informar a Amazon GameLift de su estado. Para activar la comunicación con Amazon GameLift, añada código a su proyecto de servidor de juegos. Para obtener más información, consulte Integración de juegos con servidores de juegos personalizados.

1. Prepare el servidor de juegos personalizado para el alojamiento en Amazon GameLift.
   - Obtenga el SDK del servidor de Amazon GameLift y compílelo para el lenguaje de programación y motor de videojuegos que desee.
   - Para activar la comunicación con Amazon GameLift, añada código a su proyecto de servidor de juegos.
2. Prepare el cliente de juegos para conectarlo a sesiones de juego alojadas en Amazon GameLift.
   - Añada el SDK de AWS al servicio de backend y al proyecto del cliente de juegos. Para obtener más información, consulte Descarga de los SDK de Amazon GameLift para los servicios de cliente.
   - Añada la funcionalidad para recuperar información sobre las sesiones de juego, crear sesiones de juego nuevas y reservar espacio para los jugadores en una sesión de juego.
   - Utilice FlexMatch para el emparejamiento de jugadores (opcional). Para obtener más información, consulte Integración de FlexMatch con el alojamiento de Amazon GameLift.

Preparación del servidor de Realtime

Servidores en tiempo real de Amazon GameLift proporciona una solución de servidores ligera que puede configurar para adaptarlos a su juego. Un servidor de Realtime ofrece las mismas ventajas que ofrece Amazon GameLift a los servidores de juegos, pero con una capacidad de personalización del servidor de juegos reducida.

Cree un script de Realtime para alojarlo en Amazon GameLift.

Los scripts de Realtime contienen la configuración del servidor y la lógica de juego personalizada (opcional). Los servidores de Realtime están diseñados para iniciar y detener sesiones de juego, aceptar conexiones de jugadores y administrar la comunicación con Amazon GameLift y entre los jugadores de un juego. También hay enlaces para añadir una lógica de servidor personalizada al juego. Los servidores de Realtime utilizan Node.js y JavaScript. Para obtener más información, consulte Creación de un script de Realtime y Realización de una prueba de integración con Amazon GameLift.

Realización de una prueba de integración con Amazon GameLift

Amazon GameLift admite iteraciones rápidas cuando se prueban los servidores de juegos. En este tema se tratarán los tipos de pruebas disponibles.

Servidores de juegos personalizados

Utilice Amazon GameLift para integrar hardware Anywhere en su entorno en la arquitectura de alojamiento de juegos de Amazon GameLift. Amazon GameLift Anywhere registra el hardware con Amazon GameLift en una flota de Anywhere para que pueda probarlo con su propio equipo de desarrollo local. Para obtener más información sobre la realización de pruebas con Amazon GameLift Anywhere, consulte Realización de una prueba de integración con flotas de Amazon GameLift Anywhere. Para obtener más información sobre el uso de Amazon GameLift Anywhere para alojar sus juegos con soluciones en las instalaciones, consulte Elección de los recursos GameLift informáticos de Amazon.

Realtime Servers

Con Realtime Servers, podrá actualizar sus scripts en cualquier momento. Al actualizar un script de Realtime, Amazon GameLift distribuirá la nueva versión a sus recursos de alojamiento en cuestión de minutos. Una vez que Amazon GameLift implemente el nuevo script, todas las sesiones de juego nuevas utilizarán la nueva versión del script. Cuando Amazon GameLift implemente el nuevo script, podrá empezar a realizar las pruebas inmediatamente. Para obtener más información sobre Realtime Servers, consulte Integración de juegos con Servidores en tiempo real de Amazon GameLift.

Planificación e implementación de los recursos de Amazon GameLift

Siga los consejos que se proporcionan a continuación para planificar la implementación global de los recursos de Amazon GameLift. Para obtener información sobre dónde puede alojar sus juegos con Amazon GameLift, consulte Ubicaciones GameLift de alojamiento de Amazon.

Para implementar sus recursos de Amazon GameLift, realice las siguientes tareas:

- Empaquetar y cargar su servidor de juegos en Amazon GameLift o en su hardware. Cuando cargue el servidor en Amazon GameLift, cárguelo solo en la Región de AWS de origen de su flota. Amazon GameLift distribuye automáticamente el servidor a otras ubicaciones de la flota. Para obtener más información, consulte Carga de compilaciones y scripts en Amazon GameLift.
- Diseñar e implementar una flota de Amazon GameLift para su juego. Determine el tipo de recursos informáticos que se va a utilizar, en qué ubicaciones se va a implementar, si desea utilizar colas y otras opciones. Para obtener más información, consulte Guía de diseño de flotas de Amazon GameLift.

- Crear colas para administrar las nuevas ubicaciones de las sesiones de juego y las estrategias de instancias de spot. Para obtener más información, consulte Diseño de colas de sesiones de juego.
- Utilizar el escalado automático para administrar la capacidad de alojamiento de la flota según la demanda prevista de los jugadores. Para obtener más información, consulte Escalación de la capacidad de alojamiento de Amazon GameLift.
- Usar las reglas de emparejamiento de FlexMatch para su juego. Para obtener más información, consulte Integración de FlexMatch con el alojamiento de Amazon GameLift.

Implementación automática de los recursos de Amazon GameLift

Para optimizar la implementación global de sus recursos de Amazon GameLift, le recomendamos que utilice la infraestructura como código (IaC) para definir los recursos. Puesto que Amazon GameLift admite plantillas de AWS CloudFormation, puede establecer parámetros en las plantillas para cualquier configuración específica de la implementación.

Para administrar la implementación de sus pilas de AWS CloudFormation, también le recomendamos utilizar herramientas y servicios de integración y entrega continuas (CI/CD), como AWS CodePipeline. Esos recursos le ayudarán con la implementación automática y la aprobación cada vez que cree un archivo binario para el servidor de juegos.

Los siguientes son algunos pasos habituales de la implementación de recursos de Amazon GameLift para una nueva versión del servidor de juegos que puede automatizar mediante una herramienta o un servicio de CI/CD:

- Compilación y realización de pruebas del archivo binario de su servidor de juegos.
- Carga del archivo binario en Amazon GameLift o en el hardware.
- Implementación de nuevas flotas en la nueva compilación.
- Eliminación de las flotas de la versión anterior de la cola de Amazon GameLift y reemplazo por las flotas nuevas después de implementarlas.
- Eliminación de las pilas de AWS CloudFormation de las flotas de la versión anterior una vez que esas flotas finalicen correctamente todas las sesiones de juego.

También puede utilizar AWS Cloud Development Kit (AWS CDK) para definir sus recursos de Amazon GameLift. Para obtener más información acerca del AWS CDK, consulte la Guía para desarrolladores del AWS Cloud Development Kit (AWS CDK).

Diseño del servicio de cliente de juegos

Le recomendamos que implemente un servicio de cliente de juegos que autentique a sus jugadores y se comunique con la API de Amazon GameLift. Al implementar un servicio de cliente de juegos personalizado, podrá realizar las siguientes acciones:

- Personalizar la autenticación de sus jugadores..
- Controlar la forma en que Amazon GameLift empareja e inicia las sesiones de juego.
- Utilizar la base de datos de jugadores para conocer los atributos de los jugadores, como su nivel de habilidad para realizar emparejamientos en lugar de confiar en el cliente.

Si se usa un servicio de cliente de juegos, también se reducen los riesgos de seguridad que presentan los clientes de juegos que interactúan directamente con la API de Amazon GameLift.

Autenticación de los jugadores

Puede utilizar Amazon Cognito y los ID de sesión de los jugadores para autenticar los clientes de juego. Para administrar el ciclo de vida y las propiedades de las identidades de sus jugadores, utilice los grupos de usuarios de Amazon Cognito.

Si lo prefiere, cree una solución de identidad personalizada y alójela en AWS. También puede utilizar autorizadores Lambda para una lógica de autorización personalizada con API Gateway.

Recursos adicionales:

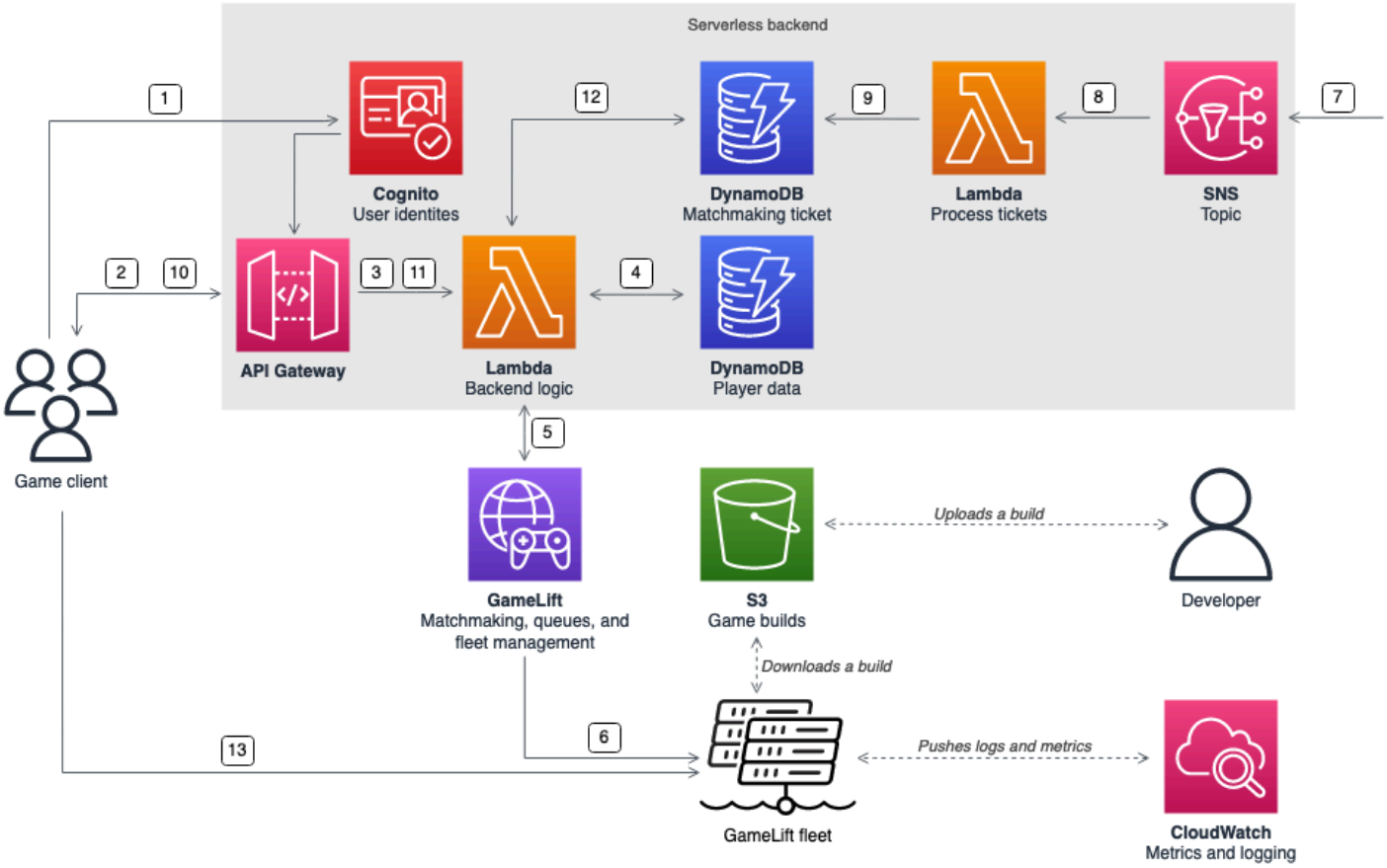
- Uso de grupos de identidades (identidades federadas) (Guía para desarrolladores de Amazon Cognito)
- Introducción a los grupos de usuarios (Guía para desarrolladores de Amazon Cognito)
- Cómo configurar la autenticación del jugador con Amazon Cognito (AWS para el blog de juegos)

Servidores de sesión de juego independientes con un backend sin servidor

Al utilizar una arquitectura de servicio de cliente sin servidor, el backend puede ver el estado de los tickets de emparejamiento desde una base de datos altamente escalable en lugar de acceder directamente a la API de Amazon GameLift.

En el siguiente diagrama se muestra un backend sin servidor creado con Servicios de AWS que empareja jugadores con juegos que se ejecutan en las flotas de Amazon GameLift. La siguiente lista

proporciona una descripción de cada aviso numerado en el diagrama. Para probar este ejemplo, consulte Alojamiento de juegos multijugador basados en sesiones en AWS en GitHub.



1. El cliente del juego solicita una identidad de usuario de Amazon Cognito de un grupo de identidades de Amazon Cognito.
2. El cliente del juego recibe credenciales de acceso temporales y solicita una sesión de juego a través de una API de Amazon API Gateway.
3. API Gateway invoca una función AWS Lambda.
4. La función de Lambda solicita los datos del reproductor de una tabla NoSQL de Amazon DynamoDB. La función proporciona la identidad de Amazon Cognito en los datos de contexto de la solicitud.
5. La función de Lambda solicita una coincidencia mediante el emparejamiento de Amazon GameLift FlexMatch.
6. FlexMatch empareja a un grupo de jugadores con la latencia adecuada y, a continuación, solicita la ubicación de una sesión de juego a través de una cola de Amazon GameLift. La cola tiene flotas con una o más ubicaciones de Región de AWS en ella.

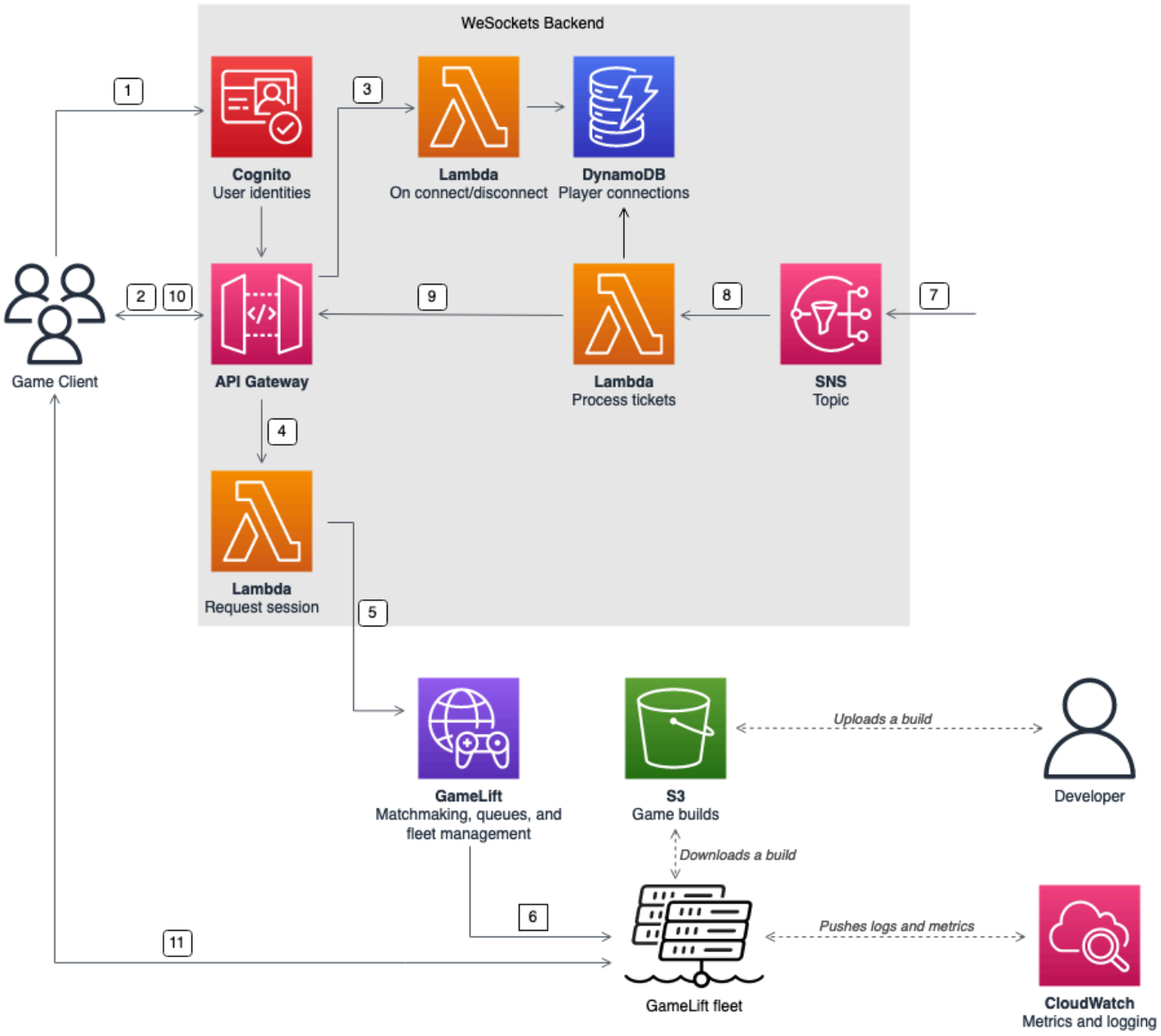
7. Cuando Amazon GameLift coloca la sesión en una de las ubicaciones de la flota, Amazon GameLift envía una notificación de evento a un tema de Amazon Simple Notification Service (Amazon SNS).
8. Una función de Lambda recibe el evento de Amazon SNS y lo procesa.
9. Si el ticket de emparejamiento es un evento `MatchmakingSucceeded`, la función de Lambda escribe el resultado, junto con el puerto y la dirección IP del servidor de juegos, en una tabla de DynamoDB.
10. El cliente del juego envía una solicitud firmada a API Gateway para ver el estado del ticket de emparejamiento en un intervalo específico.
11. API Gateway utiliza una función de Lambda que comprueba el estado del ticket de emparejamiento.
12. La función de Lambda comprueba la tabla de DynamoDB para comprobar si el ticket es correcto. Si lo es, la función devuelve al cliente el puerto y la dirección IP del servidor de juegos, junto con el ID de sesión del jugador. Si el ticket no es correcto, la función envía una respuesta verificando que el emparejamiento no está listo aún.
13. El cliente del juego se conecta al servidor de juegos mediante TCP o UDP mediante el puerto y la dirección IP que proporciona el servicio de backend. A continuación, el cliente del juego envía el ID de sesión del jugador al servidor de juegos, que lo valida mediante el SDK del servidor de Amazon GameLift.

Servidores de sesión de juego independientes con un backend basado en WebSocket

Utilice una arquitectura basada en WebSocket de Amazon API Gateway para realizar solicitudes de emparejamiento con WebSockets y enviar notificaciones push para completar el emparejamiento mediante mensajes iniciados por el servidor. Esta arquitectura mejora el rendimiento mediante una comunicación bidireccional entre el cliente y el servidor.

Para obtener más información sobre las API de WebSocket de API Gateway, consulte Trabajar con API de WebSocket.

En el siguiente diagrama se muestra una arquitectura de backend basada en WebSocket que utiliza API Gateway y otros Servicios de AWS para emparejar a los jugadores con los juegos que se ejecutan en las flotas de Amazon GameLift. La siguiente lista proporciona una descripción de cada aviso numerado en el diagrama.



1. El cliente del juego solicita una identidad de usuario de Amazon Cognito de un grupo de identidades de Amazon Cognito.
2. El cliente del juego firma una conexión de WebSocket a una API de API Gateway con las credenciales de Amazon Cognito.
3. API Gateway llama a una función AWS Lambda de la conexión. La función almacena la información de conexión en una tabla de Amazon DynamoDB.
4. El cliente del juego envía un mensaje a una función de Lambda, a través de la API de API Gateway a través de la conexión WebSocket para solicitar una sesión.

5. Una función de Lambda recibe el mensaje y, a continuación, solicita un emparejamiento mediante el emparejamiento de Amazon GameLift FlexMatch.
6. Después, FlexMatch empareja a un grupo de jugadores y FlexMatch solicita la ubicación de una sesión de juego a través de una cola de Amazon GameLift.
7. Cuando Amazon GameLift coloca la sesión en una de las ubicaciones de la flota, Amazon GameLift envía una notificación de evento a un tema de Amazon Simple Notification Service (Amazon SNS).
8. Una función de Lambda recibe el evento de Amazon SNS y lo procesa.
9. Si el ticket de emparejamiento es un evento `MatchmakingSucceeded`, la función de Lambda solicita a DynamoDB la conexión de jugador correcta. A continuación, la función envía un mensaje al cliente de juegos mediante la API de API Gateway a través de la conexión WebSocket. En esta arquitectura, el cliente del juego no consulta activamente el estado del emparejamiento.
10. El cliente del juego recibe el puerto y la dirección IP del servidor de juegos, junto con el ID de sesión del jugador, a través de la conexión WebSocket.
11. El cliente del juego se conecta al servidor de juegos mediante TCP o UDP mediante el puerto y la dirección IP que proporciona el servicio de backend. A continuación, el cliente del juego envía el ID de sesión del jugador al servidor de juegos, que lo valida mediante el SDK del servidor de Amazon GameLift.

Configuración de métricas y registros para Amazon GameLift

Puede utilizar datos recopilados de sus servidores y recursos de juegos de Amazon GameLift para ayudar a identificar anomalías. También puede usar métricas para mejorar el rendimiento.

Entre las áreas clave que tener en cuenta para Amazon GameLift se incluyen las siguientes:

- Métricas del servicio de Amazon GameLift: Amazon GameLift proporciona métricas de Amazon CloudWatch sobre sus recursos, incluidos los servidores de juegos, las flotas, las colas y FlexMatch. También puede encontrar esas métricas en las consolas de Amazon GameLift y CloudWatch. Para obtener más información sobre las métricas de Amazon GameLift en CloudWatch, consulte Supervisión de Amazon GameLift con Amazon CloudWatch.
- Métricas del servidor de juegos: Amazon GameLift no puede acceder a las métricas del servidor de juegos. Sin embargo, puede enviar métricas personalizadas a CloudWatch directamente desde el servidor de juegos a través del agente de CloudWatch. También puede utilizar el rol de AWS Identity and Access Management (IAM) de la flota y el SDK de AWS para enviar métricas

directamente a CloudWatch. Para ver un ejemplo de cómo configurar las métricas, consulte Alojamiento de juegos multijugador basados en sesiones en AWS en GitHub. Este repositorio incluye un ejemplo de configuración y código del agente de CloudWatch para un cliente de StatsD de C#.

- Registros del servidor de juegos: para configurar los archivos de registro del servidor de juegos en el servidor de juegos, utilice la configuración del SDK del servidor de Amazon GameLift. También puede usar Registros de Amazon CloudWatch como una solución de administración de registros en tiempo real y configurar los registros con el agente de CloudWatch. Para obtener más información, consulte Registro de mensajes del servidor en Amazon GameLift.

Listas de verificación para el lanzamiento de juegos

Puede utilizar estas listas de verificación para validar las fases de implementación del juego. En las listas de verificación, los elementos marcados como [Crítico] son fundamentales para el lanzamiento a la producción.

Temas

Incorporación

Utilice la siguiente lista de verificación para realizar un seguimiento de los elementos para incorporar el juego al alojamiento de Amazon GameLift. Los elementos marcados como [Crítico] son fundamentales para el lanzamiento a la producción.

- [Critico] Complete el cuestionario de incorporación de Amazon GameLift en la consola Amazon GameLift.
- [Crítico] Diseñe e implemente un servicio de backend para que los clientes de juego interactúen con sus servidores de juegos.
- [Crítico] Cree AWS Identity and Access Managementroles (IAM) que proporcionen a las instancias del servidor de Amazon GameLift acceso otros recursos de AWS.

- [Crítico] Diseñe e implemente la conmutación por error a otras Regiones de AWS para FlexMatch y colas.
- Planifique la implementación de las flotas en las ubicaciones de destino teniendo en cuenta las colas y la estructura de las flotas del juego.
- Automatice la implementación mediante la infraestructura como código (IaC) con AWS CloudFormation y AWS Cloud Development Kit (AWS CDK).
- Recopile registros y análisis con Amazon CloudWatch y Amazon Simple Storage Service (Amazon S3).

Pruebas

Utilice la siguiente lista de verificación para realizar un seguimiento de los elementos de prueba mientras desarrolla el juego con el alojamiento de Amazon GameLift. Los elementos marcados como [Crítico] son fundamentales para el lanzamiento a la producción.

- [Crítico] Complete el cuestionario de lanzamiento y envíelo completo al equipo de lanzamiento de Amazon GameLift. Puede encontrar el cuestionario de lanzamiento en la consola de Amazon GameLift.
- [Crítico] Solicite aumentos en las cuotas de servicio de Amazon GameLift y otras cuotas de Servicio de AWS para que el entorno en vivo pueda escalar verticalmente las necesidades de producción.
- [Crítico] Verifique que los puertos abiertos de las flotas activas coincidan con el rango de puertos que podrían utilizar sus servidores.
- [Crítico] Cierre el puerto RDP 3389 y el puerto SSH 22.
- Desarrolle un plan para la gestión de DevOps de su juego. Si utiliza Amazon CloudWatch Logs o las métricas personalizadas de Amazon CloudWatch, defina alarmas para problemas graves o críticos en la flota de servidores. Simule los errores y pruebe los manuales de ejecución.
- Verifique que la cantidad de servidores que se ejecutan en una instancia en pleno uso esté dentro de las capacidades del tipo de instancia de servidor.
- Ajuste su política de escalado para que sea más conservadora al principio y proporcione más capacidad inactiva de la que cree que necesita. Puede optimizar los costos más adelante. Considere el uso de una política de escalado basada en objetivos con una capacidad inactiva del 20 por ciento.

- Utilice las reglas de latencia de FlexMatch para emparejar a los jugadores que se encuentran geográficamente cerca de la misma Región de AWS. Compruebe cómo se comporta bajo carga con datos de latencia sintéticos del cliente de pruebas de carga.
- Ponga a prueba su infraestructura de autenticación de jugadores y sesiones de juego para comprobar si se escala de forma eficaz a la demanda.
- Verifique que un servidor que ha estado funcionando durante varios días siga aceptando conexiones.
- Aumente el nivel del plan de AWS Support Business o Enterprise para que AWS pueda responderle en caso de problemas o interrupciones.

Lanzar

Utilice la siguiente lista de verificación para realizar un seguimiento de los elementos de lanzamiento para su juego alojado en Amazon GameLift. Los elementos marcados como [Crítico] son fundamentales para el lanzamiento a la producción.

- [Crítico] Configure la política de protección de la flota para proteger por completo todas las flotas activas, de modo que la reducción vertical no interrumpa las sesiones de juego activas.
- [Crítico] Establezca un tamaño máximo de flota lo suficientemente alto como para adaptarse a los picos de demanda previstos, como mínimo. Le recomendamos que duplique el tamaño máximo para una demanda no prevista.
- Anime a todo el equipo de desarrollo a participar en el evento de lanzamiento y a supervisar el lanzamiento del juego en una sala de lanzamiento.
- Supervise la latencia y la experiencia de los jugadores.

posterior al lanzamiento

Utilice la siguiente lista de verificación para realizar un seguimiento de los elementos posteriores al lanzamiento para su juego alojado en Amazon GameLift.

- Ajuste las reglas de escalado para minimizar la capacidad de inactividad.
- Modifique las reglas de FlexMatch o añada ubicaciones adicionales en función de sus requisitos de latencia.

- Optimice el servidor ejecutable, ya que la eficiencia de su rendimiento afecta directamente a los costos de la flota. Para ejecutar más sesiones de juego con la misma infraestructura, aumente la cantidad de procesos del servidor por instancia.
- Utilice los datos de análisis para impulsar el desarrollo continuo, mejorar la experiencia de los jugadores y la longevidad del juego, y optimizar la monetización.

Preparando juegos para Amazon GameLift

Para preparar tu juego multijugador para alojarlo en Amazon GameLift, configura la comunicación entre tu juego y Amazon GameLift. Los temas de esta sección proporcionan ayuda detallada para integrar tu juego con Amazon GameLift, servidores de juegos personalizados y servidores en tiempo real, y para añadir matchmaking con. FlexMatch

Temas

Integración de juegos con servidores de juegos personalizados

Amazon GameLift proporciona un conjunto de herramientas completo para preparar los juegos multijugador y servidores de juegos personalizados para que se ejecuten en Amazon GameLift. Los SDK de Amazon GameLift contienen las bibliotecas necesarias para que los clientes y servidores de juegos se comuniquen con Amazon GameLift. Para obtener más información sobre los SDK y dónde obtenerlos, consulte Soporte de desarrollo con Amazon GameLift.

Los temas de esta sección contienen instrucciones detalladas sobre cómo añadir la funcionalidad de Amazon GameLift para el cliente de juego y el servidor de juegos antes de la implementación en Amazon GameLift. Si desea ver una hoja de ruta completa para poner a punto su juego y ejecutarlo con Amazon GameLift, consulte Hoja de ruta de alojamiento administrado de Amazon GameLift.

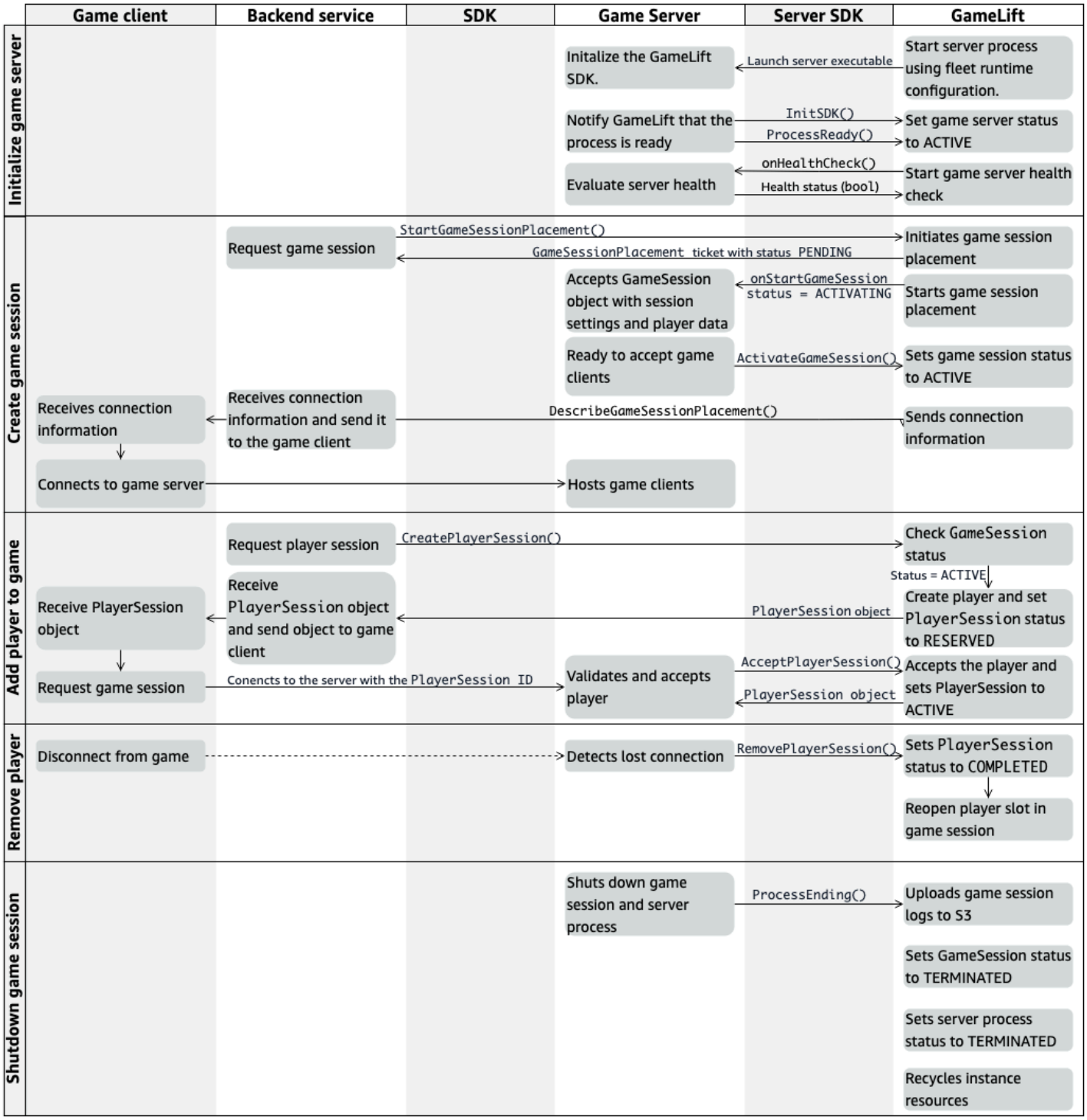
Temas

- Realización de una prueba de integración con Amazon GameLift Local

Interacciones entre Amazon GameLift y el servidor de cliente del juego

En este tema se describen las interacciones entre un cliente de juegos, un servicio de backend, un servidor de juegos y Amazon GameLift.

El siguiente diagrama muestra las interacciones entre el cliente de juegos, el servicio de backend, el SDK de Amazon GameLift, el servidor de juegos EC2 administrado, el SDK del servidor de Amazon GameLift y Amazon GameLift. Para obtener una descripción detallada de las interacciones mostradas, consulte las siguientes secciones de esta página.



Inicialización de un servidor de juegos

Los siguientes pasos describen las interacciones que se producen al preparar el servidor de juegos para albergar sesiones de juego.

1. Amazon GameLift inicia el archivo ejecutable del servidor en una instancia de Amazon Elastic Compute Cloud (Amazon EC2) (Amazon EC2).
2. El servidor de juegos llama a los siguientes elementos:

   a. `InitSDK()` para inicializar el SDK del servidor.

   b. `ProcessReady()` para comunicar que la sesión de juego está lista, la información de la conexión y la ubicación de los archivos de registro de la sesión de juego.

   A continuación, el proceso del servidor espera a que Amazon GameLift le devuelva la llamada.
3. Amazon GameLift actualiza el estado del proceso del servidor a `ACTIVE` para habilitar la ubicación de las sesiones de juego.
4. Amazon GameLift comienza a solicitar la devolución de llamada de `onHealthCheck` y continúa realizando la llamada periódicamente mientras el proceso del servidor está activo. El proceso del servidor puede informar sobre el estado en el plazo de un minuto.

Crear una sesión de juego.

Una vez inicializado el servidor de juegos, se producen las siguientes interacciones al crear sesiones de juego para alojar a los jugadores.

1. El servicio de backend llama a la operación del SDK `StartGameSessionPlacement()`.
2. Amazon GameLift crea un nuevo ticket `GameSessionPlacement` con el estado `PENDING` y lo devuelve al servicio de backend.
3. El servicio de backend obtiene el estado de un ticket de ubicación a partir de una cola. Para obtener más información, consulte Configuración de la notificación de eventos para la ubicación de sesiones de juego..
4. Amazon GameLift inicia la ubicación de las sesiones de juego mediante la selección una flota adecuada y la búsqueda de un proceso de servidor activo en una flota con `0` sesiones de juego. Cuando Amazon GameLift localiza un proceso de servidor, Amazon GameLift realiza el siguiente procedimiento:

   a. Crea un objeto `GameSession` con la configuración de la sesión del juego y los datos del jugador a partir de la solicitud de ubicación con un estado `ACTIVATING`.

   b. Invoca la devolución de llamada de `onStartGameSession` del proceso del servidor. Amazon GameLift transfiere información al objeto `GameSession` que indica que el proceso del servidor puede configurar la sesión de juego.

   c. Cambia el número de sesiones de juego del proceso del servidor a `1`.

5. El proceso del servidor ejecuta la función de devolución de llamada de `onStartGameSession`. Cuando el proceso del servidor está listo para aceptar las conexiones del jugador, llama a `ActivateGameSession()` y espera las conexiones de los jugadores.
6. Amazon GameLift actualiza el objeto `GameSession` con información de conexión para el proceso del servidor (esa información incluye la configuración de puerto de la que se informó con `ProcessReady()`). Amazon GameLift también cambia el estado a `ACTIVE`.
7. El servicio de backend llama a `DescribeGameSessionPlacement()` para detectar el estado actualizado del ticket. A continuación, el servicio de backend utiliza la información de conexión para conectar el cliente del juego con el proceso del servidor y unirse a la sesión de juego.

Adición de un jugador a un juego

Esta secuencia describe el proceso de adición de un jugador a una sesión de juego existente. También se pueden solicitar sesiones de jugador como parte de una solicitud de ubicación de sesión de juego.

1. El servicio de backend llama a la operación de la API del cliente `CreatePlayerSession()` con un ID de sesión de juego.
2. Amazon GameLift comprueba el estado de la sesión de juego (debe ser `ACTIVE`) y busca una ranura de jugador abierta en la sesión de juego. Si hay una ranura disponible, Amazon GameLift realiza el siguiente procedimiento:

   a. Crea un objeto `PlayerSession` nuevo y establece el estado en `RESERVED`.

   b. Responde a la solicitud de servicio de backend con el objeto `PlayerSession`.

3. El servicio de backend conecta el cliente de juegos directamente con el proceso del servidor con el ID de sesión de jugador.
4. El servidor llama a la operación de la API de servidor `AcceptPlayerSession()` para validar el ID de sesión de jugador. Si se valida, Amazon GameLift transfiere el objeto `PlayerSession` al proceso del servidor. A continuación, el proceso del servidor acepta o rechaza la conexión.
5. Amazon GameLift realiza uno de los siguientes procedimientos:

a. Si se acepta la conexión, Amazon GameLift establece el estado `PlayerSession` en `ACTIVE`.

b. Si no se recibe respuesta en un plazo de 60 segundos de la llamada `CreatePlayerSession()` original del servidor de backend, Amazon GameLift cambia el estado de `PlayerSession` a `TIMEDOUT` y vuelve a abrir la ranura de jugador en la sesión de juego.

Eliminación de un jugador

Cuando se eliminan jugadores de una sesión de juego para crear espacio para que se unan nuevos jugadores, se producen las siguientes interacciones.

1. Un jugador se desconecta del juego.
2. El servidor detecta la pérdida de conexión y llama a la acción de la API de servidor `RemovePlayerSession()`.
3. Amazon GameLift cambia el estado `PlayerSession` a `COMPLETED` y vuelve a abrir la ranura del jugador en la sesión de juego.

Cierre de la sesión de juego

Esta secuencia de interacciones se produce cuando un proceso del servidor cierra la sesión de juego actual.

1. El servidor cierra la sesión de juego y el servidor.
2. El servidor llama a `ProcessEnding()` en Amazon GameLift.
3. Amazon GameLift realiza el siguiente procedimiento:

   a. Carga registros de sesión de juego en Amazon Simple Storage Service (Amazon S3).

   b. Cambia el estado `GameSession` a `TERMINATED`.

   c. Cambia el estado del proceso del servidor a `TERMINATED`.

   d. Recicla los recursos de la instancia.

Integración del servidor de juegos con Amazon GameLift

Una vez que el servidor de juegos personalizado se haya implementado y ejecutado en las instancias de Amazon GameLift, debe poder interactuar con Amazon GameLift (y posiblemente con otros recursos). En esta sección se describe cómo integrar su software de servidor de juegos con Amazon GameLift

> **Note**
>
> Estas instrucciones dan por sentado que ha creado una Cuenta de AWS y que ya tiene un proyecto de servidor de juegos.

Los siguientes temas describen cómo administrar las siguientes tareas de integración:

- Establecer la comunicación entre Amazon GameLift y tus servidores de juegos.
- Generar y utilizar un certificado TLS para establecer una conexión segura entre el cliente y el servidor de juegos.
- Proporcionar permisos para que el software del servidor de juegos interactúe con otros recursos de AWS.
- Permitir que los procesos del servidor de juegos obtengan información sobre la flota en la que se ejecutan.

Temas

Adición de Amazon GameLift al servidor de juegos

Su servidor de juegos personalizado debe comunicarse con Amazon GameLift, ya que cada proceso del servidor de juegos debe poder responder a los eventos que inicie Amazon GameLift. El servidor de juegos también debe mantener a Amazon GameLift informado sobre el estado del proceso del servidor y las conexiones de los jugadores. Para obtener más información sobre cómo el servidor de juegos, el servicio de backend, el cliente del juego y Amazon GameLift trabajan juntos para administrar el alojamiento de juegos, consulte Interacciones entre Amazon GameLift y el servidor de cliente del juego.

Para preparar el servidor de juegos para que interactúe con Amazon GameLift, añada el SDK de servidor de Amazon GameLift en el proyecto del servidor de juegos y compile la funcionalidad descrita en este tema. El SDK del servidor está disponible en varios idiomas. Para obtener más información sobre el SDK del servidor de Amazon GameLift, consulte Soporte de desarrollo con Amazon GameLift.

Referencias de la API del SDK del servidor:

- Referencia del SDK del servidor Amazon GameLift 5.x para C++
- Referencia del SDK del servidor de Amazon GameLift 5.x para C# y Unity
- Referencia del SDK del servidor de Unreal Engine de Amazon GameLift 5.x

Inicialización del proceso del servidor

Añada código para establecer la comunicación con Amazon GameLift e informar de que el proceso del servidor está listo para alojar una sesión de juego. Este código debe ejecutarse antes que cualquier código de Amazon GameLift.

1. Llame a `InitSdk()` para iniciar el cliente de la API de Amazon GameLift. Para inicializar un proceso de servidor en un recurso informático de Amazon GameLift Anywhere, llame a `InitSdk()` con los siguientes `ServerParameters`:

   - La URL del WebSocket que se utiliza para conectarse al servidor de juegos.
   - El ID del proceso utilizado para alojar su servidor de juegos.
   - El ID del proceso utilizado para alojar los procesos del servidor de juegos.
   - El ID de la flota de GameLift que contiene el entorno informático de Amazon GameLift Anywhere.
   - El token de autorización generado por la operación de Amazon GameLift `GetComputeAuthToken`.

   > **Note**
   >
   > Para iniciar un servidor de juegos en una instancia de Amazon EC2 administrada de Amazon GameLift, cree sus ServerParameters mediante el constructor predeterminado `InitSDK()` (C++) (C#) (Unreal) (sin parámetros). Amazon GameLift configura el entorno informático y se conecta automáticamente a Amazon GameLift por usted.

2. Notifique a Amazon GameLift que el proceso del servidor de juegos está listo para alojar una sesión de juego. Llame a `ProcessReady()` (C++) (C#) (Unreal) con la siguiente información. Tenga en cuenta que debe llamar a `ProcessReady()` solo una vez por proceso del servidor.

   - El número de puerto que utiliza el proceso del servidor. El servicio de backend proporciona el número de puerto y una dirección IP a los clientes del juego para que se conecten al proceso del servidor y se unan a una sesión de juego.
   - La ubicación de los archivos, como los registros de sesiones de juegos, que quiere que Amazon GameLift retenga. El proceso del servidor genera esos archivos durante una sesión de juego. Se almacenan temporalmente en la instancia en la que se ejecuta el proceso del servidor y se pierden cuando la instancia se cierra. Todos los archivos que publique se cargarán en Amazon GameLift. Puede acceder a los archivos a través de la consola de Amazon GameLift o llamando a la operación de la API de Amazon GameLift GetGameSessionLogUrl().
   - Se procesarán los nombres de las funciones de devolución de llamadas a las que Amazon GameLift puede llamar en su servidor. El servidor de juegos debe implementar las siguientes funciones. Para obtener más información, consulte (C++) (C#) (Unreal) .
     - `onHealthCheck` (opcional): Amazon GameLift llama a esta función con frecuencia para solicitar al servidor un informe del estado.
     - `onStartGameSession`: Amazon GameLift llama a esta función como respuesta a la solicitud de cliente CreateGameSession().
     - `onProcessTerminate`: Amazon GameLift fuerza la detención del proceso del servidor, lo que permite que se cierre sin problemas.
     - `onUpdateGameSession` (opcional): Amazon GameLift ofrece un objeto de sesión de juego actualizado al servidor de juegos o proporciona una actualización de estado de una solicitud de reposición de emparejamiento. La característica Reposición de FlexMatch requiere esta devolución de llamada.

   También puede configurar un servidor de juegos para acceder de forma segura a los recursos de AWS que le pertenecen o que controla. Para obtener más información, consulte Comunicación con otros recursos de AWS de sus flotas.

Notificación del estado del proceso del servidor (opcional)

Añada código a su servidor de juegos para implementar la función de devolución de llamada `onHealthCheck()`. Amazon GameLift invoca este método de devolución de llamadas periódicamente para recopilar métricas de estado. Al implementar esta función de devolución de llamada, realice el siguiente procedimiento:

- Evalúe el estado del proceso del servidor. Por ejemplo, puede informar de que el proceso del servidor no está en buen estado si alguna dependencia externa ha fallado.
- Complete la evaluación del estado y responda a la devolución de llamada en un plazo de 60 segundos. Si Amazon GameLift no recibe una respuesta durante ese periodo, considerará automáticamente que el proceso del servidor no está funcionando correctamente.
- Devuelva un valor booleano: true para buen estado y false para mal estado.

Si no implementa una devolución de llamada de comprobación de estado, Amazon GameLift considerará que el proceso del servidor está en buen estado a menos que el servidor no responda.

Amazon GameLift utiliza el estado de los procesos del servidor para finalizar los procesos que no funcionan correctamente y liberar recursos. Si se sigue informando de que un proceso del servidor no funciona correctamente o no responde a tres comprobaciones de estado consecutivas, Amazon GameLift puede cerrar el proceso e iniciar uno nuevo. Amazon GameLift recopila métricas sobre el estado de los procesos de los servidores de una flota.

Obtención de un certificado TLS (opcional)

Si el proceso del servidor se ejecuta en una flota que tiene activada la generación de certificados TLS, puede recuperar el certificado TLS para establecer una conexión segura con un cliente de juegos y cifrar la comunicación cliente/servidor. En la instancia se almacena una copia del certificado. Para obtener la ubicación del archivo, llame a `GetComputeCertificate()` (C++) (C#) (Unreal) .

Inicio de una sesión de juego

Añada código para implementar la función de devolución de llamada `onStartGameSession`. Amazon GameLift invoca esta devolución de llamada para iniciar una sesión de juego en el servidor.

La función `onStartGameSession` toma un objeto GameSession como parámetro de entrada. Este objeto incluye información clave de la sesión de juego, como el número máximo de jugadores. También puede incluir datos del juego y de los jugadores. Durante el proceso de implementación de la función se deben realizar las siguientes tareas:

- Inicie acciones para crear una nueva sesión de juego en función de las propiedades de `GameSession`. Como mínimo, el servidor de juegos debe asociar el ID de sesión del juego, al que los clientes del juego hacen referencia cuando se conectan al proceso del servidor.
- Procese los datos del juego y de los jugadores según sea necesario. Esos datos se encuentran en el objeto `GameSession`.
- Informe a Amazon GameLift cuando haya una nueva sesión de juego lista para aceptar jugadores. Llame a la operación de la API del servidor `ActivateGameSession()`(C++) (C#) (Unreal) (C++)Unreal). En respuesta a una llamada realizada correctamente, Amazon GameLift cambia el estado de la sesión de juego a `ACTIVE`.

Validación de un jugador nuevo (opcional)

Si realiza un seguimiento del estado de las sesiones de los jugadores, añada un código para validar un jugador nuevo cuando se conecte a un servidor de juegos. Amazon GameLift realiza un seguimiento de los jugadores actuales y de las ranuras de sesión de juego disponibles.

Para la validación, el cliente del juego que solicite acceso a la sesión de juego debe incluir un ID de sesión del jugador. Amazon GameLift genera automáticamente este ID cuando un jugador solicita unirse a un juego mediante StartGamessionPlacement() o StartMatchmaking(). La sesión del jugador reserva entonces un espacio libre en una sesión de juego.

Cuando el proceso del servidor de juegos recibe una solicitud de conexión con el cliente del juego, llama a `AcceptPlayerSession()` (C++) (C#) (Unreal) (C++) con el ID de sesión del jugador. Como respuesta, Amazon GameLift comprueba que el ID de sesión del jugador se corresponde con una ranura abierta reservada en la sesión de juego. Una vez que Amazon GameLift valida el ID de sesión del jugador, el proceso del servidor acepta la conexión. A continuación, el jugador puede unirse a la sesión de juego. Si Amazon GameLift no valida el ID de sesión del jugador, el proceso del servidor rechaza la conexión.

Notificación de la finalización de una sesión de jugador (opcional)

Si está realizando un seguimiento del estado de las sesiones de los jugadores, añada un código para informar a Amazon GameLift cuando un jugador abandone la sesión de juego. Este código debe ejecutarse siempre que el proceso del servidor detecte la interrupción de una conexión. Amazon GameLift utiliza esta notificación para realizar un seguimiento de los jugadores actuales y de las ranuras disponibles en la sesión de juego.

Para gestionar las conexiones interrumpidas, añada en el código una llamada a la operación de la API del servidor `RemovePlayerSession()` (C++) (C#) (Unreal) (C++) con el ID de sesión del jugador correspondiente.

Finalización de una sesión de juego

Añada código a la secuencia de cierre del proceso del servidor para informar a Amazon GameLift cuando finalice una sesión de juego. Para reciclar y actualizar los recursos de alojamiento, Amazon GameLift cierra los procesos del servidor una vez finalizada la sesión de juego.

Al iniciar el código de cierre del proceso del servidor, llame a la operación de la API del servidor `ProcessEnding()` (C++) (C#) (Unreal) . Esta llamada informa a Amazon GameLift de que el proceso del servidor se va a apagar. Amazon GameLift cambia el estado de la sesión del juego y el estado del proceso del servidor a `TERMINATED`. Después de llamar a `ProcessEnding()`, es seguro que el proceso se cierre.

Respuesta a una notificación de cierre del proceso del servidor

Añada un código para cerrar el proceso del servidor como respuesta a una notificación de Amazon GameLift. Amazon GameLift envía eta notificación cuando el proceso del servidor informa constantemente de que no funciona correctamente o si la instancia en la que se está ejecutando el proceso del servidor está finalizando. Amazon GameLift puede detener una instancia como parte de un evento de reducción vertical de la capacidad o como respuesta a una interrupción de una instancia de spot.

Para gestionar una notificación de cierre, realice los siguientes cambios en el código del servidor de juegos:

- Implemente la función de devolución de llamada on`ProcessTerminate()`. Esta función debe llamar al código que apaga el proceso del servidor. Cuando Amazon GameLift invoca esta operación, las interrupciones de instancias de spot se notifican con dos minutos de antelación. El aviso proporciona al proceso del servidor tiempo para desconectar a los jugadores de forma correcta, mantener los datos de estado del juego y realizar otras tareas de limpieza.
- Llame a la operación de la API del servidor `GetTerminationTime()` (C++) (C#) (Unreal) desde el código de cierre del servidor de juegos. Si Amazon GameLift ha emitido una llamada para detener el proceso del servidor, `GetTerminationTime()` devuelve el tiempo de finalización estimado.
- Al iniciar el código de cierre del proceso del servidor de juegos, llame a la operación de la API del servidor `ProcessEnding()` (C++) (C#) (Unreal) . Esta llamada informa a Amazon GameLift de

que el proceso del servidor se está cerrando y, a continuación, Amazon GameLift cambia el estado del proceso del servidor a `TERMINATED`. Después de llamar a `ProcessEnding()`, es seguro que el proceso se cierre.

Comunicación con otros recursos de AWS de sus flotas

Al crear una compilación de servidor de juegos para implementarla en las flotas de Amazon GameLift, es posible que desee que las aplicaciones de la compilación de juegos se comuniquen de forma directa y segura con otros recursos de AWS de su propiedad. Puesto que Amazon GameLift administra sus flotas de alojamiento de juegos, debe conceder a Amazon GameLift un acceso limitado a estos recursos y servicios.

Entre los ejemplos de escenarios se incluyen los siguientes:

- Utilización de un agente de Amazon CloudWatch para recopilar métricas, registros y seguimientos de flotas de Anywhere y flotas de EC2 administradas
- Envío de datos de registro de las instancias a Amazon CloudWatch Logs.
- Obtención de archivos de juegos almacenados en un bucket de Amazon Simple Storage Service (Amazon S3).
- Lectura y escritura de datos de juegos (como el inventario o los modos de juego) almacenados en una base de datos de Amazon DynamoDB u otro servicio de almacenamiento de datos.
- Envío de señales directamente a una instancia mediante Amazon Simple Queue Service (Amazon SQS).
- Acceso a recursos personalizados implementados y que se ejecutan en Amazon Elastic Compute Cloud (Amazon EC2).

Amazon GameLift admite los siguientes métodos para establecer el acceso:

- Acceso a los recursos de AWS con un rol de IAM
- Acceda a recursos de AWS con el emparejamiento de VPC.

Acceso a los recursos de AWS con un rol de IAM

Utilice un rol de IAM para especificar quién puede acceder a sus recursos y establecer límites a ese acceso. Las partes de confianza pueden «asumir» un rol y obtener credenciales de seguridad

temporales que les autoricen a interactuar con los recursos. Cuando las partes realizan solicitudes de la API relacionadas con el recurso, deben incluir las credenciales.

Para configurar el acceso controlado por un rol de IAM, lleve a cabo las siguientes tareas:

1. Creación del rol de IAM
2. Modificación de las aplicaciones para adquirir credenciales
3. Asociación del rol de IAM a una flota

Creación del rol de IAM

En este paso, cree un rol de IAM, con un conjunto de permisos para controlar el acceso a sus recursos de AWS y una política de confianza que otorgue a Amazon GameLift derechos para usar los permisos del rol.

Para obtener instrucciones sobre cómo configurar el rol de IAM, consulte Configurar un rol de servicio de IAM para Amazon GameLift. Al crear la política de permisos, elija servicios, recursos y acciones específicos con los que deben trabajar sus aplicaciones. Como práctica recomendada, limite el ámbito de los permisos tanto como sea posible.

Después de crear el rol, anote el nombre de recurso de Amazon (ARN) del rol. El ARN del rol será necesario durante la creación de la flota.

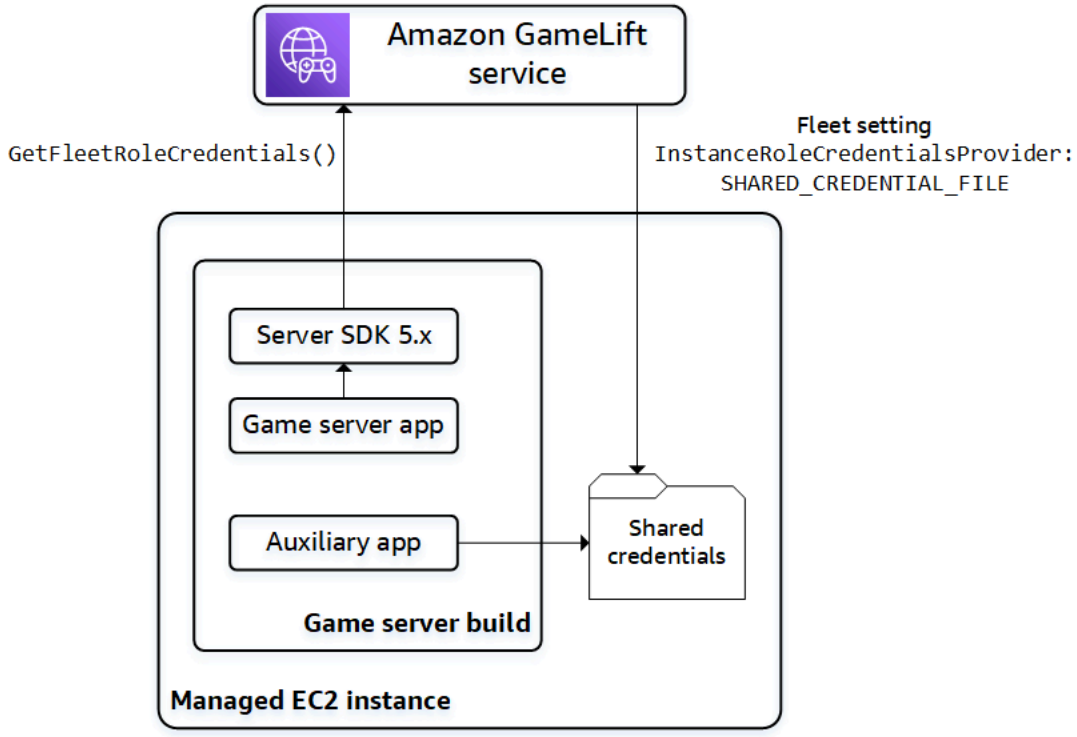
Modificación de las aplicaciones para adquirir credenciales

En este paso, debe configurar las aplicaciones para que adquieran credenciales de seguridad para el rol de IAM y las utilicen al interactuar con sus recursos de AWS. Consulte la siguiente tabla para determinar cómo modificar las aplicaciones en función de (1) el tipo de aplicación y (2) la versión del SDK del servidor que utilice el juego para comunicarse con Amazon GameLift.

	Aplicaciones del servidor de juegos	Otras aplicaciones
Uso del SDK del servidor, versión 5.x	Llame al método del SDK del servidor `GetFleetRoleCredentials()` desde el código del servidor de juegos.	Añada código a la aplicación para obtener las credenciales de un archivo compartido en la instancia de la flota.

	Aplicaciones del servidor de juegos	Otras aplicaciones
Uso de la versión 4 o anterior del SDK del servidor	Llame a AWS Security Token Service (AWS STS) `AssumeRole` con el ARN del rol.	Llame a AWS Security Token Service (AWS STS) `AssumeRole` con el ARN del rol.

En el caso de los juegos integrados con el SDK 5.x del servidor, este diagrama muestra cómo las aplicaciones de la compilación del juego implementada pueden adquirir credenciales para el rol de IAM.



Llamada a **`GetFleetRoleCredentials()`** (SDK del servidor 5.x)

En el código del servidor de juegos, que ya debería estar integrado con el SDK del servidor de Amazon GameLift 5.x, llame a `GetFleetRoleCredentials` (C++) (C#) (Unreal) para recuperar un conjunto de credenciales temporales. Cuando las credenciales caduquen, puede actualizarlas con otra llamada a `GetFleetRoleCredentials`.

Utilización de credenciales compartidas (SDK del servidor 5.x)

En el caso de las aplicaciones que no son de servidor y que se implementan con compilaciones de servidores de juegos que utilizan el SDK de servidor 5.x, añade código para obtener y utilizar

las credenciales almacenadas en un archivo compartido. Amazon GameLift genera un perfil de credenciales para cada instancia de flota. Las credenciales están disponibles para que las utilicen todas las aplicaciones de la instancia. Amazon GameLift actualiza continuamente las credenciales temporales.

Debe configurar una flota para generar el archivo de credenciales compartidas al crear la flota.

En cada aplicación que necesite utilizar el archivo de credenciales compartido, especifique la ubicación del archivo y el nombre del perfil de la siguiente manera:

Windows:

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

Linux:

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

Ejemplo: Configuración de un agente de CloudWatch para recopilar métricas para las instancias de flota de Amazon GameLift

Si desea utilizar un agente de Amazon CloudWatch para recopilar métricas, registros y seguimientos de sus flotas de Amazon GameLift, utilice este método para autorizar al agente a emitir los datos a su cuenta. En este escenario, realice los pasos siguientes:

1. Recupere o escriba el archivo `config.json` del agente de CloudWatch.
2. Actualice el archivo `common-config.toml` del agente para identificar el nombre del archivo de credenciales y el nombre del perfil, tal y como se describió previamente.
3. Configure el script de instalación de la compilación del servidor de juegos para instalar e iniciar el agente de CloudWatch.

Utilización de **`AssumeRole()`** (SDK del servidor 4)

Añada código a sus aplicaciones para asumir el rol de IAM y obtener credenciales para interactuar con sus recursos de AWS. Cualquier aplicación que se ejecute en una instancia de flota de Amazon GameLift con un SDK del servidor 4 o anterior puede asumir el rol de IAM.

En el código de la aplicación, antes de acceder a un recurso de AWS, la aplicación debe llamar a la operación de la API `AssumeRole` API AWS Security Token Service (AWS STS) y especificar el ARN del rol. Esta operación devuelve un conjunto de credenciales temporales que autoriza a la aplicación a tener acceso al recurso de AWS. Para obtener más información, consulte Uso de credenciales temporales con AWS en la Guía del usuario de IAM.

Asociación del rol de IAM a una flota

Una vez que haya creado el rol de IAM y actualizado las aplicaciones de la compilación del servidor de juegos para obtener y usar las credenciales de acceso, podrá implementar una flota. Al configurar la nueva flota, establezca los siguientes parámetros:

- InstanceRoleArn: establezca ese parámetro en el ARN del rol de IAM.
- InstanceRoleCredentialsProvider: para solicitar a Amazon GameLift que genere un archivo de credenciales compartido para cada instancia de flota, establezca el parámetro en `SHARED_CREDENTIAL_FILE`.

Debe establecer esos valores al crear la flota. No se podrán actualizar más tarde.

Acceda a recursos de AWS con el emparejamiento de VPC.

Puede utilizar el emparejamiento de Amazon Virtual Private Cloud (Amazon VPC) para comunicarse entre aplicaciones que se ejecutan en una instancia de Amazon GameLift y otro recurso de AWS. Una VPC es una red virtual privada definida por usted que incluye un conjunto de recursos que se administran a través de su Cuenta de AWS. Cada flota de Amazon GameLift tiene su propia VPC. Gracias al emparejamiento de VPC, puede establecer una conexión de red directa entre la VPC de una flota y de otros recursos de AWS.

Amazon GameLift simplifica el proceso de configuración de las conexiones de emparejamiento de VPC para los servidores de juegos. Gestiona las solicitudes de interconexión, actualiza la tablas de ruteo y configura las conexiones según sea necesario. Para obtener instrucciones sobre cómo establecer el emparejamiento de VPC para los servidores de juegos, consulte Emparejamiento de VPC para Amazon GameLift.

Integración del cliente de juegos con Amazon GameLift

En los temas de esta sección se describe la funcionalidad administrada de Amazon GameLift que puede añadir a un servicio de backend. El servicio de backend se encarga de las siguientes tareas:

- Solicita información sobre las sesiones de juego activas de Amazon GameLift.

- Conecta a un jugador a una sesión de juego existente.
- Crea una nueva sesión de juego y une los jugadores a ella.
- Cambia los metadatos de una sesión de juego existente.

Para obtener más información sobre cómo interactúan los clientes de juego con Amazon GameLift y los servidores de juegos que se ejecutan en Amazon GameLift, consulte Interacciones entre Amazon GameLift y el servidor de cliente del juego.

Requisitos previos

- Una Cuenta de AWS.
- Una compilación de servidor de juegos cargada en Amazon GameLift.
- Una flota para alojar los juegos.

Temas

- Añade Amazon GameLift a tu cliente de juegos
- Generación de ID de jugador

Añade Amazon GameLift a tu cliente de juegos

Integra Amazon GameLift en los componentes del juego que necesitan información sobre la sesión del juego, crea nuevas sesiones de juego y añade jugadores a los juegos. Dependiendo de la arquitectura del juego, esta funcionalidad se coloca en servicios de backend que administran tareas como la autenticación de los jugadores, el emparejamiento o la ubicación de la sesión de juego.

> **Note**
>
> Para obtener información detallada sobre cómo configurar el matchmaking para tu juego GameLift alojado en Amazon, consulta la Guía para GameLift FlexMatch desarrolladores de Amazon.

Configurar Amazon GameLift en un servicio de back-end

Agrega código para inicializar un GameLift cliente de Amazon y almacenar la configuración clave. Este código debe ejecutarse antes que cualquier código que dependa de Amazon GameLift.

1. Configure una configuración del cliente. Utilice la configuración de cliente predeterminada o cree un objeto de configuración de cliente personalizado. Para obtener más información, consulte AWS::Client::ClientConfiguration(C++) o AmazonGameLiftConfig(C#).

   La configuración de un cliente especifica la región y el punto final de destino que se utilizarán al contactar con Amazon GameLift. La región identifica el conjunto de recursos implementados (flotas, colas y emparejadores) que se van a utilizar. La configuración del cliente predeterminada establece la ubicación en la región Este de EE. UU. (Norte de Virginia). Para utilizar cualquier otra región, cree una configuración personalizada.

2. Inicializa un GameLift cliente de Amazon. Utilice Aws:GameLift:: GameLiftClient () (C++) o AmazonGameLiftClient() (C#) con una configuración de cliente predeterminada o personalizada.

3. Añada un mecanismo para generar un identificador único para cada jugador. Para obtener más información, consulte Generación de ID de jugador.

4. Recopile y almacene la siguiente información:

   - Flota objetivo: muchas solicitudes de GameLift API de Amazon deben especificar una flota. Para ello, utilice un ID de la flota o un ID de alias que apunte hacia la flota de destino. Como práctica recomendada, utilice los alias de flota para poder cambiar los jugadores de una flota a otra sin tener que actualizar los servicios de backend.
   - Cola de destino: en el caso de los juegos que utilizan colas de varias flotas para organizar nuevas sesiones de juego, especifique el nombre de la cola que se va a utilizar.
   - AWS credenciales: todas las llamadas a Amazon GameLift deben proporcionar las Cuenta de AWS credenciales del anfitrión del juego. Estas credenciales se obtienen mediante la creación de un usuario jugador, tal y como se describe en Configuración de acceso mediante programación para su juego. En función de cómo administre el acceso del usuario jugador, realice los siguientes procedimientos:
     - Si utilizas un rol para gestionar los permisos de usuario de los jugadores, añade código para asumir el rol antes de llamar a una GameLift API de Amazon. La solicitud para asumir el rol devolverá un conjunto de credenciales de seguridad temporales. Para obtener más información, consulte Cambiar a un rol de IAM (AWS API) en la Guía del usuario de IAM.
     - Si dispone de credenciales de seguridad de larga duración, configure el código para localizar y utilizar las credenciales almacenadas. Consulte Autenticación mediante credenciales de larga duración en la Guía de referencia de SDK y herramientas de AWS . Para obtener información sobre el almacenamiento de credenciales, consulte las referencias de las AWS API para (C++) y (.NET).

- Si dispone de credenciales de seguridad temporales, añada código para actualizarlas periódicamente mediante AWS Security Token Service (AWS STS), tal y como se describe en la sección Uso de credenciales de seguridad temporales con los SDK de AWS en la Guía del usuario de IAM. El código debe solicitar nuevas credenciales antes de que las antiguas caduquen.

Obtención de sesiones de juego

Añada código para descubrir las sesiones de juego disponibles y administrar la configuración de las sesiones de juego y los metadatos.

Búsqueda de sesiones de juego activas

Se usa SearchGameSessionspara obtener información sobre una sesión de juego específica, todas las sesiones activas o las sesiones que cumplen un conjunto de criterios de búsqueda. Esta llamada devuelve un GameSessionobjeto por cada sesión de juego activa que coincide con tu solicitud de búsqueda.

Utilice los criterios de búsqueda para obtener una lista filtrada de las sesiones de juego activas a las que puedan conectarse los jugadores. Por ejemplo, puede filtrar sesiones de la siguiente manera:

- Excluya las sesiones de juego llenas: `CurrentPlayerSessionCount = MaximumPlayerSessionCount`.
- Elija las sesiones de juego en función de la duración de la sesión: Evaluar `CreationTime`.
- Encuentre sesiones de juego en función de una propiedad de juego personalizada: `gameSessionProperties.gameMode = "brawl"`

Administración de sesiones de juego

Utilice cualquiera de las operaciones siguientes para recuperar o actualizar la información de la sesión de juego.

- DescribeGameSessionDetails() — Obtén el estado de protección de una sesión de juego además de la información sobre la sesión de juego.
- UpdateGameSession() — Cambia los metadatos y la configuración de una sesión de juego según sea necesario.
- GetGameSessionLogUrl— Accede a los registros de sesiones de juego almacenados.

Creación de sesiones de juego

Añada código para iniciar sesiones de juego nuevas en las flotas implementadas y ponerlas a disposición de los jugadores. Hay dos opciones para crear sesiones de juego, dependiendo de si vas a desplegar el juego en varias regiones Regiones de AWS o en una sola.

Creación de una sesión de juego en una cola de varias ubicaciones

Se usa StartGameSessionPlacementpara poner en cola una solicitud para una nueva sesión de juego. Para usar esta operación de, cree una cola. Esto determina dónde GameLift coloca Amazon la nueva sesión de juego. Para obtener más información sobre las colas y cómo utilizarlas, consulte Configuración de colas de Amazon GameLift para la ubicación de las sesiones de juego.

Al crear una ubicación de la sesión de juego, especifique el nombre de la cola que utilizar, un nombre de sesión de juego, la cantidad máxima de jugadores simultáneos y un conjunto de propiedades del juego opcionales. Si lo desea, también puede proporcionar una lista de los jugadores para que se unan automáticamente a la sesión de juego. Si incluyes los datos de latencia de los jugadores de las regiones pertinentes, Amazon GameLift utilizará esta información para incluir la nueva sesión de juego en una flota que ofrezca la experiencia de juego ideal para los jugadores.

La ubicación de sesiones de juego es un proceso asíncrono. Una vez realizada la solicitud, puede esperar a que funcione o a que caduque. También puedes cancelar la solicitud en cualquier momento utilizando StopGameSessionPlacement. Para comprobar el estado de su solicitud de colocación, llame DescribeGameSessionPlacement.

Creación de una sesión de juego en una flota específica

Se usa CreateGameSessionpara crear una nueva sesión en una flota específica. Esta operación síncrona funciona o no dependiendo de si la flota dispone de los recursos necesarios para alojar una sesión de juego nueva. Cuando Amazon GameLift cree la nueva sesión de juego y devuelva un GameSessionobjeto, podrás unir a los jugadores a ella.

Si utiliza esta operación, proporcione un ID de la flota o de alias, un nombre de sesión y un número máximo de jugadores simultáneos para dicho juego. De forma opcional, puede incluir un conjunto de propiedades del juego. Las propiedades del juego se definen en una matriz de pares clave-valor.

Si utilizas la función de protección de GameLift recursos de Amazon para limitar el número de sesiones de juego que puede crear un jugador, proporciona el ID de jugador del creador de la sesión de juego.

Conexión de un jugador a una sesión de juego

Añada código para reservar una ranura de jugador en una sesión de juego activa y conectar clientes de juego a sesiones de juego.

1. Reserva de una ranura de jugador en una sesión de juego

   Para reservar una ranura de jugador, cree una sesión de jugador nueva para la sesión de juego. Para obtener más información sobre las sesiones de jugador, consulte Cómo se conectan los jugadores a los juegos.

   Existen dos formas de crear sesiones de jugador nuevas:

   - Se usa StartGameSessionPlacementpara reservar espacios para uno o más jugadores en la nueva sesión de juego.
   - Reserva espacios para uno o más jugadores que usen CreatePlayerSessiono CreatePlayerSessionstengan un identificador de sesión de juego.

   Amazon GameLift primero verifica que la sesión de juego acepte nuevos jugadores y que tenga espacios disponibles para jugadores. Si tiene éxito, Amazon GameLift reserva un espacio para el jugador, crea la nueva sesión de jugador y devuelve un PlayerSessionobjeto. Este objeto contiene el nombre de DNS, la dirección IP y el puerto que necesita el cliente de juegos para conectarse a la sesión de juego.

   Una solicitud de sesión de jugador debe incluir un ID exclusivo para cada jugador. Para obtener más información, consulte Generación de ID de jugador.

   Una sesión de jugador puede incluir un conjunto de datos de jugador personalizados. Estos datos se almacenan en el objeto de sesión del jugador recién creado, que puedes recuperar llamando a DescribePlayerSessions(). Amazon GameLift también pasa este objeto al servidor del juego cuando el jugador se conecta directamente a la sesión de juego. A la hora de solicitar varias sesiones de jugador, proporcione una cadena de datos de jugador para cada jugador asignada al ID de jugador en la solicitud.

2. Conexión a una sesión de juego

   Añada código al cliente de juego para recuperar el objeto `PlayerSession`, que contiene la información de conexión de la sesión de juego. Utilice esta información para establecer una conexión directa con el servidor.

- Puede conectarse mediante el puerto especificado y el nombre de DNS o la dirección IP asignados al proceso del servidor.
- Si las flotas tienen habilitada la generación de certificados TLS, conéctese mediante el nombre de DNS y el puerto.
- Si el servidor de juegos valida las conexiones de los jugadores entrantes, consulte el ID de sesión del jugador.

Tras realizar la conexión, el proceso del cliente y el servidor del juego se comunican directamente sin la intervención de Amazon GameLift. El servidor mantiene comunicación con Amazon GameLift para informar sobre el estado de conexión de los jugadores, su estado de salud y más. Si el servidor de juegos valida a los jugadores entrantes, verifica que el ID de sesión del jugador coincide con una ranura reservada en la sesión de juego y acepta o rechaza la conexión del jugador. Cuando el jugador se desconecte, el proceso del servidor notificará que se ha desconectado.

Usa las propiedades de la sesión del juego

El cliente de juego puede transferir datos a una sesión de juego mediante una propiedad del juego. Las propiedades del juego son pares clave-valor que el servidor de juegos puede añadir, leer, enumerar y cambiar. Puedes transferir una propiedad del juego al crear una nueva sesión de juego o más adelante, cuando la sesión de juego esté activa. Una sesión de juego puede contener hasta 16 propiedades de juego. No puedes eliminar las propiedades del juego.

Por ejemplo, tu juego ofrece los siguientes niveles de dificultad: `NoviceEasy`,`Intermediate`, y`Expert`. El jugador elige y`Easy`, a continuación, comienza el juego. El cliente del juego solicita una nueva sesión de juego a Amazon GameLift utilizando una de las secciones anteriores `StartGameSessionPlacement` o `CreateGameSession` tal como se explica en ellas. En la solicitud, el cliente pasa esto:`{"Key": "Difficulty", "Value":"Easy"}`.

En respuesta a la solicitud, Amazon GameLift crea un `GameSession` objeto que contiene la propiedad del juego especificada. GameLift Luego, Amazon indica a un servidor de juegos disponible que inicie la nueva sesión de juego y pasa el `GameSession` objeto. El servidor del juego inicia una sesión de juego con un `Difficulty` de`Easy`.

Más información

- GameProperty tipo de datos

- SearchGameSessions() ejemplos
- UpdateGameSession GameProperties parámetro ()

Generación de ID de jugador

Amazon GameLift utiliza una sesión de jugador para representar un jugador conectado a una sesión de juego. Amazon GameLift crea una sesión de jugador cada vez que un jugador se conecta a una sesión de juego mediante un cliente de juego integrado con Amazon GameLift. Cuando un jugador sale de un juego, la sesión de jugador finaliza. Amazon GameLift no reutiliza las sesiones de los jugadores.

El siguiente ejemplo de código genera ID de jugador únicos de forma aleatoria:

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

Para obtener más información sobre las sesiones de jugador, consulte Visualización de datos de sesiones de juego y de jugador.

Motores de juegos y Amazon GameLift

Puede utilizar el servicio de Amazon GameLift administrado con la mayoría de motores de videojuegos compatibles con bibliotecas C++ o C#, incluidos O3DE, Unreal Engine y Unity. Compile la versión que necesite para el juego; consulte los archivos README de cada versión para ver las instrucciones de compilación y conocer los requisitos mínimos. Para obtener más información sobre los SDK de Amazon GameLift disponibles, las plataformas de desarrollo y los sistemas operativos admitidos, consulte Soporte de desarrollo con Amazon GameLift para ver los servidores de juegos.

Además de la información específica del motor proporcionada en este tema, encontrará más ayuda para la integración de Amazon GameLift en los servidores, clientes y servicios de juegos en los siguientes temas:

- Hoja de ruta de alojamiento administrado de Amazon GameLift: un flujo de trabajo de seis pasos para integrar correctamente Amazon GameLift en el juego y configurar los recursos de alojamiento.
- Adición de Amazon GameLift al servidor de juegos: instrucciones detalladas sobre la integración de Amazon GameLift en un servidor de juegos.

- Añade Amazon GameLift a tu cliente de juegos: instrucciones detalladas sobre la integración en un cliente o servicio de juego, incluida la creación de sesiones de juego y la conexión de jugadores a juegos.

O3DE

Servidores de juegos

Prepare los servidores de juegos para su alojamiento en Amazon GameLift mediante el SDK del servidor de Amazon GameLift para C++. Consulte Adición de Amazon GameLift al servidor de juegos para obtener ayuda con la integración de la funcionalidad necesaria en el servidor de juegos.

Clientes y servicios de juego

Active los clientes o servicios de juegos para interactuar con el servicio de Amazon GameLift, por ejemplo, para buscar sesiones de juego disponibles o para crear sesiones nuevas y agregar jugadores a los juegos. Consulte el SDK de AWS para C++. para obtener información sobre las funcionalidades principales del cliente. Para integrar Amazon GameLift en el proyecto de juego de O3DE, consulte Adición de Amazon GameLift a un cliente y servidor de juegos de O3DE y Añade Amazon GameLift a tu cliente de juegos.

Unreal Engine

Servidores de juegos

Prepare los servidores de juegos para el alojamiento en Amazon GameLift añadiendo el SDK del servidor de Amazon GameLift para Unreal Engine al proyecto e implementando la funcionalidad de servidor necesaria. Para obtener ayuda para configurar el complemento de Unreal Engine y añadir el código de Amazon GameLift, consulte Integre Amazon GameLift en un proyecto de Unreal Engine.

Clientes y servicios de juego

Active los clientes o servicios de juegos para interactuar con el servicio de Amazon GameLift, por ejemplo, para buscar sesiones de juego disponibles o para crear sesiones nuevas y agregar jugadores a los juegos. Consulte el SDK de AWS para C++. para obtener información sobre las funcionalidades principales del cliente. Para integrar Amazon GameLift en el proyecto de juego de Unreal Engine, consulte Añade Amazon GameLift a tu cliente de juegos.

Unity

Servidores de juegos

Prepare los servidores de juegos para el alojamiento en Amazon GameLift añadiendo el SDK del servidor de Amazon GameLift para C# al proyecto e implementando la funcionalidad de servidor necesaria. Para obtener ayuda sobre la configuración con Unity y la adición de código de Amazon GameLift, consulte Integración de Amazon GameLift en un proyecto de Unity.

Clientes y servicios de juego

Active los clientes o servicios de juegos para interactuar con el servicio de Amazon GameLift, por ejemplo, para buscar sesiones de juego disponibles o para crear sesiones nuevas y agregar jugadores a los juegos. Consulte el AWS SDK for .NET para obtener información sobre las funcionalidades principales del cliente. Para integrar Amazon GameLift en el proyecto de juego de Unity, consulte Añade Amazon GameLift a tu cliente de juegos.

Otros motores

Si desea obtener una lista completa de los SDK de Amazon GameLift disponibles para servidores y clientes de juegos, consulte the section called “Admisión de entornos de desarrollo”.

Adición de Amazon GameLift a un cliente y servidor de juegos de O3DE

Puede utilizar O3DE, un motor 3D de código abierto, multiplataforma y en tiempo real para crear experiencias interactivas de alto rendimiento, incluidos juegos y simulaciones. El renderizador y las herramientas de O3DE están integrados en un marco modular que puede modificar y ampliar con sus herramientas de desarrollo preferidas.

El marco modular utiliza gemas que contienen bibliotecas con interfaces y activos estándar. Seleccione sus propias gemas para elegir qué funcionalidad añadir en función de sus necesidades.

La gema Amazon GameLift ofrece las siguientes características:

Integración con Amazon GameLift

> Un marco para ampliar la capa de red de O3DE y permitir que la gema multijugador funcione con la solución de servidor dedicado de Amazon GameLift. La gema ofrece integraciones tanto con el SDK del servidor de Amazon GameLift como con el cliente del SDK de AWS (para llamar al propio servicio de Amazon GameLift).

Administración de compilaciones y paquetes

Instrucciones para empaquetar y, opcionalmente, cargar la compilación del servidor dedicado y una aplicación AWS Cloud Development Kit (AWS CDK) (AWS CDK) para configurar y actualizar los recursos.

Configuración de la gema Amazon GameLift

Siga los procedimientos de esta sección para configurar la gema Amazon GameLift en O3DE.

Requisitos previos

- Configure su cuenta de AWS en Amazon GameLift. Para obtener más información, consulte Configura un Cuenta de AWS.
- Defina las credenciales de AWS para O3DE. Para obtener más información, consulte Configuración de credenciales de AWS.
- Configure la AWS CLI y el AWS CDK. Para obtener más información, consulte la AWS Command Line Interface y el AWS Cloud Development Kit (AWS CDK).

Activación de la gema Amazon GameLift y sus dependencias

1. Abra el Administrador de proyectos.
2. Abra el menú del proyecto y elija Editar configuración del proyecto....
3. Elija Configurar gemas.
4. Active la gema Amazon GameLift y las siguientes gemas dependientes:
   - Gema principal de AWS: proporciona el marco que utilizarán los Servicios de AWS en O3DE.
   - Gema multijugador: proporciona la funcionalidad multijugador al ampliar el marco de red.

Inclusión de la biblioteca estática de la gema Amazon GameLift

1. Incluya el `Gem::AWSGameLift.Server.Static` como `BUILD_DEPENDENCIES` para el destino del servidor de su proyecto.

```
ly_add_target(
    NAME YourProject.Server.Static STATIC
    ...
```

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

2. Configure el `AWSGameLiftService` como obligatorio para el componente del sistema del servidor de proyectos.

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

3. Para realizar solicitudes de servicio de Amazon GameLift en C++, incluya `Gem::AWSGameLift.Client.Static` en `BUILD_DEPENDENCIES` para su el destino de cliente (opcional).

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

Integración del juego y del servidor dedicado

Administre las sesiones de juego en el juego y en el servidor de juegos dedicado con la función de integración de administración de sesiones. Para admitir FlexMatch, consulte Integración de FlexMatch.

Integre Amazon GameLift en un proyecto de Unreal Engine

En este tema se explica cómo configurar el complemento SDK del servidor Amazon GameLift C++ para Unreal Engine e integrarlo en tus proyectos de juegos.

Recursos adicionales:

- Complemento del SDK de servidor para el sitio de descargas de Unreal
- Referencia del SDK del servidor de Unreal Engine de Amazon GameLift 5.x
- the section called “Admisión de entornos de desarrollo”

Requisitos previos

Antes de continuar, asegúrese de comprobar que se cumplen los siguientes requisitos previos:

Requisitos previos

- Un ordenador con capacidad para ejecutar Unreal Engine. Para obtener más información sobre los requisitos de Unreal Engine, consulte la documentación Especificaciones de hardware y software de Unreal Engine.
- Microsoft Visual Studio 2019 o una versión posterior.
- Versión 3.1 o posterior de CMake.
- Versión 3.6 o posterior de Python.
- Un cliente Git disponible en PATH.
- Una cuenta de Epic Games. Cree una cuenta en el sitio web oficial de Unreal Engine.
- Una GitHub cuenta asociada a tu cuenta de Unreal Engine. Para obtener más información, consulta Cómo acceder al código fuente de Unreal Engine en el GitHub sitio web de Unreal Engine.

> **Note**
>
> Amazon GameLift actualmente admite las siguientes versiones de Unreal Engine:
>
> - 4.22
> - 4.23
> - 4.24

- 4.25
- 4.26
- 4.27
- 5.1.0
- 5.1.1
- 5.2
- 5.3

Compilación de Unreal Engine a partir del código fuente

Las versiones estándar del editor de Unreal Engine, descargadas a través del lanzador de Epic solo permiten compilar aplicaciones cliente de Unreal. Para compilar una aplicación de servidor de Unreal, debe descargar y compilar Unreal Engine a partir del código fuente mediante el repositorio de Github de Unreal Engine. Para obtener más información, consulte el tutorial https://docs.unrealengine.com/5.1/building-unreal-engine-from-source/ en el sitio web de documentación de Unreal Engine.

> **Note**
>
> Si aún no lo has hecho, sigue las instrucciones que aparecen en Acceder al código fuente de Unreal Engine GitHub para vincular tu GitHub cuenta a tu cuenta de Epic Games.

Clonación del código fuente de Unreal Engine en su entorno de desarrollo

1. Clone el código fuente de Unreal Engine en su entorno de desarrollo en la ramificación que elija.

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

2. Consulte la etiqueta de la versión que está utilizando para desarrollar el juego. Por ejemplo, en el siguiente ejemplo se muestra la versión 5.1.1 de Unreal Engine:

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

3. Diríjase a la carpeta raíz del repositorio local. Cuando esté en la carpeta raíz, ejecute el siguiente archivo:`Setup.bat`.
4. Mientras esté en la carpeta raíz, ejecute también el archivo: `GenerateProjectFiles.bat`.

5. Después de ejecutar los archivos de los pasos anteriores, se creará un archivo de solución de Unreal Engine, `UE5.sln`. Abra Visual Studio y, en el editor de Visual Studio, abra el archivo `UE5.sln`.
6. En Visual Studio, abra el menú Ver y elija la opción Explorador de soluciones. De esa forma, se abrirá el menú contextual del nodo del proyecto de Unreal. En la ventana del Explorador de soluciones, haga clic con el botón derecho en el archivo `UE5.sln` (puede aparecer como `UE5`) y, a continuación, seleccione Compilar para compilar el proyecto de Unreal con el objetivo Win64 del editor de desarrollo.

   > **Note**
   >
   > Para completar la compilación se precisa más de una hora.

Una vez completada la compilación, estará listo para abrir el editor de desarrollo de Unreal y crear o importar un proyecto.

Configuración de un proyecto de Unreal para el complemento

Sigue estos pasos para preparar el plugin Amazon GameLift Server SDK para Unreal Engine para tus proyectos de servidores de juegos.

Configuración de un proyecto para el complemento

1. Con Visual Studio abierto, diríjase al panel del Explorador de soluciones y elija el archivo `UE5` para abrir el menú contextual del proyecto de Unreal. En el menú contextual, elija la opción Establecer como proyecto de inicio.
2. En la parte superior de la ventana de Visual Studio, elija Iniciar la depuración (flecha verde).

   Esta acción inicia la nueva instancia de Unreal Editor creada en código fuente. Para obtener más información sobre el uso del editor de Unreal, consulte Interfaz del editor de Unreal en el sitio web de documentación de Unreal Engine.
3. Cierre la ventana de Visual Studio que ha abierto, ya que el editor de Unreal abre otra ventana de Visual Studio que contiene el proyecto de Unreal y el proyecto de juego.
4. En el editor de Unreal, realice uno de los siguientes procedimientos:
   - Elige un proyecto de Unreal existente que quieras integrar con Amazon GameLift.
   - Cree un nuevo proyecto de . Para experimentar con el GameLift plugin de Amazon para Unreal, prueba a usar la plantilla en tercera persona del motor Unreal. Para obtener más

información sobre esta plantilla, consulte la plantilla Tercera persona en el sitio web de documentación de Unreal Engine.

También puede configurar un nuevo proyecto con la siguiente configuración:

- C++
- Con contenido inicial
- Escritorio
- Un nombre de proyecto. En los ejemplos de este tema, asignamos un nombre a nuestro proyecto `GameLiftUnrealApp`.

5. En el Explorador de soluciones de Visual Studio, diríjase a la ubicación de su proyecto de Unreal. En la carpeta `Source` de Unreal, busque un archivo denominado *`Your-application-name`*`.Target.cs`.

   Por ejemplo: `GameLiftUnrealApp.Target.cs`.

6. Realice una copia del archivo y asígnele el nombre *`Your-application-name`*`Server.Target.cs`.
7. Abra el archivo nuevo y realice los cambios siguientes:

   - Cambie los valores `class` y `constructor` para que coincidan con el nombre del archivo.
   - Cambie el valor `Type` de `TargetType.Game` a `TargetType.Server`.
   - El archivo final tendrá un aspecto semejante al siguiente:

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

Tu proyecto ahora está configurado para aceptar el complemento Amazon GameLift Server SDK.

La siguiente tarea consiste en compilar las bibliotecas del SDK del servidor C++ para Unreal de forma que pueda importarlas en su proyecto.

Para compilar las bibliotecas del SDK del servidor C++ para Unreal, realice el siguiente procedimiento:

1. Descargue el complemento SDK del servidor Amazon GameLift C++ para Unreal.

   > **Note**
   >
   > Colocar el SDK en el directorio de descargas predeterminado puede provocar un error de compilación debido a que la ruta supera el límite de 260 caracteres. Por ejemplo: `C:\Users\Administrator\Downloads\GameLift-SDK-Release-06_15_2023\GameLift-Cpp-ServerSDK-5.0.4`
   > Le recomendamos que traslade el SDK a otro directorio, por ejemplo `C:\GameLift-Cpp-ServerSDK-5.0.4`.

2. Descargue e instale OpenSSL. Para obtener más información sobre la descarga de OpenSSL, lea la documentación de compilación e instalación de OpenSSL de Github.

   Para obtener más información, lea la documentación de Notas de para plataformas Windows de OpenSSL.

   > **Note**
   >
   > La versión de OpenSSL que utilices para crear el SDK del servidor de GameLift Amazon debe coincidir con la versión de OpenSSL utilizada por Unreal para empaquetar el servidor de juegos. Puedes encontrar información sobre la versión en el directorio de instalación de Unreal. `...Engine\Source\ThirdParty\OpenSSL`

3. Con las bibliotecas descargadas, cree las bibliotecas del SDK del servidor C++ para Unreal Engine.

   En el directorio `GameLift-Cpp-ServerSDK-<version>` del SDK descargado, realice la compilación con el parámetro `-DBUILD_FOR_UNREAL=1` y compile el SDK del servidor. Los siguientes ejemplos muestran cómo realizar la compilación mediante `cmake`.

   Ejecute los siguientes comandos en el 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 compilación de Windows crea los siguientes archivos binarios en la carpeta `out\gamelift-server-sdk\Release`:

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

Copia los dos archivos de biblioteca a la `ThirdParty\GameLiftServerSDK\Win64` carpeta del paquete de complementos de Amazon GameLift Unreal Engine.

Usa el siguiente procedimiento para importar el GameLift plugin de Amazon a tu proyecto de ejemplo.

Importar el GameLift plugin de Amazon

1. Localice la `GameLiftServerSDK` carpeta que extrajo del complemento en el procedimiento anterior.
2. Ubícala `Plugins` en la carpeta raíz del proyecto del juego. (Si la carpeta no existe, créala allí).
3. Copie la `GameLiftServerSDK` carpeta en`Plugins`.

   Esto permitirá que el proyecto Unreal vea el complemento.
4. Añade el complemento SDK GameLift del servidor Amazon al `.uproject` archivo del juego.

   En el ejemplo, la aplicación se llama `GameLiftUnrealApp`, por lo que el archivo será `GameLiftUnrealApp.uproject`
5. Edite el archivo `.uproject` para añadir el complemento al proyecto de juego.

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

```
]
```

6. Asegúrate de que el juego ModuleRules dependa del complemento. Abre el `.Build.cs` archivo y añade la dependencia de Amazon GameLiftServer SDK. El archivo se encuentra en *Your-application-name*/Source/*/Your-application-name/*.

   Por ejemplo, la ruta del archivo del tutorial es `../GameLiftUnrealApp/Source/GameLiftUnrealApp/GameLiftUnrealApp.Build.cs`.

7. Añada `"GameLiftServerSDK"` al final de la lista de `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;
    }
  }
```

El complemento debería funcionar ahora para su aplicación. Continúa con la siguiente sección para integrar las GameLift funciones de Amazon en tu juego.

Agrega el código GameLift del servidor de Amazon a tu proyecto Unreal

Has configurado y configurado tu entorno de Unreal Engine y ahora puedes integrar un servidor de juegos con Amazon GameLift. El código que se presenta en este tema hace que las llamadas al GameLift servicio de Amazon sean obligatorias. También implementa un conjunto de funciones de devolución de llamadas que responden a las solicitudes del GameLift servicio de Amazon. Para obtener más información sobre cada función y lo que hace el código, consulte Inicialización del proceso del servidor. Para obtener más información sobre las acciones del SDK y los tipos de datos que se utilizan en este código, consulte Referencia del SDK del servidor de Amazon GameLift para Unreal Engine.

Para inicializar un servidor de juegos con Amazon GameLift, sigue el siguiente procedimiento.

> **Note**
>
> El código GameLift específico de Amazon que se proporciona en la siguiente sección depende del uso de un indicador de `WITH_GAMELIFT` preprocesador. Este indicador solo es válido cuando se cumplen estas dos condiciones:
>
> - `Target.Type == TargetRules.TargetType.Server`
> - Los complementos encontraron los binarios GameLift del SDK del servidor Amazon.
>
> Esto garantiza que solo las compilaciones de Unreal Server invoquen la API GameLift de backend de Amazon. También le permitirá escribir código que se ejecutará correctamente para todos los destinos diferentes de Unreal que pueda producir el juego.

Integrar un servidor de juegos con Amazon GameLift

1. En Visual Studio, abra el archivo `.sln` de su aplicación. En nuestro ejemplo, el archivo `GameLiftUnrealApp.sln` se encuentra en la carpeta raíz.
2. Con la solución abierta, localice el archivo *`Your-application-name`*`GameMode.h` de su aplicación. Ejemplo:`GameLiftUnrealAppGameMode.h`.
3. Cambie el archivo de encabezado para alinearlo con el siguiente código de ejemplo. Asegúrese de sustituir "GameLiftUnrealApp" por el nombre de su propia aplicación.

```
#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. Abra el archivo *Your-application-name*GameMode.cpp del archivo de origen relacionado. En nuestro ejemplo: GameLiftUnrealAppGameMode.cpp, y cambie el código para que se alinee con el siguiente código de ejemplo. Asegúrese de sustituir "GameLiftUnrealApp" por el nombre de su propia aplicación.

   En este ejemplo se muestra cómo añadir todos los elementos necesarios para la integración con Amazon GameLift, tal y como se describe en Añadir Amazon GameLift a tu servidor de juegos. Esto incluye:

   - Inicialización de un cliente GameLift API de Amazon.
   - Implementar funciones de devolución de llamadas para responder a las solicitudes del GameLift servicio de Amazon, incluidas OnStartGameSessionOnProcessTerminate, yonHealthCheck.
   - Llamar ProcessReady () con un puerto designado para notificar a Amazon GameLiftservice cuando esté listo para organizar sesiones de juego.

```
#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. Compile un proyecto de juego para los dos tipos de destino siguientes: Editor de desarrollo y Servidor de desarrollo.

> **Note**
>
> No es necesario volver a compilar la solución. En su lugar, compile solo el proyecto en la carpeta `Games` que coincida con el nombre de la aplicación. De lo contrario, Visual Studio volverá a compilar todo el proyecto de UE5, algo que puede tardar hasta una hora.

6. Una vez finalizadas ambas compilaciones, cierre Visual Studio y abra el archivo `.uproject` del proyecto para abrirlo en el editor de Unreal.
7. En el editor de Unreal, empaquete la compilación del servidor de juegos. Para elegir un objetivo, ve a Plataformas, Windows ***y selecciona our-application-name Servidor Y.***
8. Para iniciar el proceso de compilación de la aplicación de servidor, diríjase a Plataformas, Windows y seleccione Proyecto de paquetes. Cuando se complete la compilación, debería tener un archivo ejecutable. En el caso de nuestro ejemplo, el nombre del archivo es `GameLiftUnrealAppServer.exe`.
9. Al compilar una aplicación de servidor en el editor de Unreal, se generan dos archivos ejecutables. Uno de ellos se encuentra en la raíz de la carpeta de compilación del juego y actúa como contenedor del archivo ejecutable del servidor propiamente dicho.

   Al crear una GameLift flota de Amazon con tu versión de servidor, te recomendamos que introduzcas el ejecutable del servidor real como ruta de inicio de la configuración en tiempo de ejecución. Por ejemplo, en la carpeta de compilación del juego, puede que tenga un archivo `GameLiftFPS.exe` en la raíz y otro en `\GameLiftFPS\Binaries\Win64\GameLiftFPSServer.exe`. Al crear una flota, le recomendamos que la utilice `C:\GameLiftFPS\Binaries\Win64\GameLiftFPSServer.exe` como ruta de lanzamiento de la configuración del tiempo de ejecución.
10. Asegúrate de abrir los puertos UDP necesarios en la GameLift flota de Amazon para que el servidor del juego pueda comunicarse con los clientes del juego. De forma predeterminada, Unreal Engine utiliza el puerto `7777`. Para obtener más información, consulta UpdateFleetPortSettingsla guía de referencia de la API de Amazon GameLift Service.
11. Cree un archivo `install.bat` para la compilación del juego. Este script de instalación se ejecuta cada vez que la versión del juego se despliega en una GameLift flota de Amazon. A continuación, se muestra un archivo `install.bat` de ejemplo:

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

```
UE5PrereqSetup_x64.exe /q
```

En algunas versiones de Unreal Engine, `install.bat` debería ser

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

> **Note**
>
> La ruta del archivo al archivo `<>PrereqSetup_x64.exe` es `Engine\Extras\Redist\en-us`.

12. Ahora puedes empaquetar y subir la versión de tu juego a Amazon GameLift.

    La versión de OpenSSL que empaques con la compilación del juego debe coincidir con la versión que usó el motor del juego al crear el servidor del juego. Asegúrese de empaquetar la versión de OpenSSL correcta con la compilación del servidor de juegos. Para el SO Windows, el formato de OpenSSL es `.dll`.

    > **Note**
    >
    > Package las DLL de OpenSSL en la versión de su servidor de juegos. Asegúrate de empaquetar la misma versión de OpenSSL que usaste al crear el servidor del juego.
    >
    > - `libssl-1_1-x64.dll`
    >
    >   `libcrypto-1_1-x64.dll`
    >
    > Package sus dependencias junto con el ejecutable de su servidor de juegos en la raíz de un archivo zip. Por ejemplo, los DLL `openssl-lib` deberían estar en el mismo directorio que el archivo `.exe`.

Siguientes pasos

Has configurado y configurado tu entorno de Unreal Engine y ya puedes empezar a GameLift integrar Amazon en tu juego.

Para obtener más información sobre cómo añadir Amazon GameLift a tu juego, consulta lo siguiente:

- Adición de Amazon GameLift al servidor de juegos
- Referencia del SDK del servidor de Amazon GameLift para Unreal Engine

Para obtener instrucciones sobre cómo probar el juego, consulte Realización de una prueba de integración con flotas de Amazon GameLift Anywhere .

Integración de Amazon GameLift en un proyecto de Unity

En este tema se explica cómo configurar el complemento del SDK del servidor C# de Amazon GameLift para Unity e integrarlo en los proyectos de juegos.

Recursos adicionales:

- Sitio de descargas del SDK del servidor de Amazon GameLift
- Referencia del SDK del servidor de Amazon GameLift 5.x para C# y Unity
- the section called “Admisión de entornos de desarrollo”

Requisitos previos

Para utilizar el complemento del SDK del servidor C# de Amazon GameLift para Unity, son necesarios los siguientes componentes:

- Un entorno de desarrollo y una versión del editor Unity compatibles con el complemento (consulte Soporte de desarrollo con Amazon GameLift). Para obtener información sobre las versiones de Unity, consulte Requisitos del sistema para Unity en la documentación de Unity.
- El complemento del SDK del servidor de Amazon GameLift para el paquete Unity. Este paquete incluye el SDK del servidor 5+ para C#. Puede descargar el paquete desde este sitio: Introducción a Amazon GameLift.
- El registro de terceros se centró en UnityNuget. Esta herramienta administra archivos DLL de terceros. Para obtener más información, consulte el repositorio de Github UnityNyGet.

Configuración de UnityNuget

Si no tiene UnityNuget configurado para su proyecto de juego, siga los pasos que figuran a continuación para instalar la herramienta mediante el administrador de paquetes de Unity. También puede utilizar la CLI de NuGet para descargar manualmente los DLL. Para obtener más información, consulte el SDK del servidor C# de Amazon GameLift para el archivo `README` de Unity.

Para integrar UnityNuGet en el proyecto de juego Unity, realice el siguiente procedimiento:

1. Con el proyecto abierto en el editor de Unity, vaya al menú principal y seleccione Editar, Configuración del proyecto. Entre las opciones, elija la sección Administrador de paquetes y abra el grupo Registros con ámbito.
2. Pulse el botón + e introduzca los siguientes valores para el registro con ámbito de UnityNuget:

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

3. Para los usuarios de la versión Unity 2021:

   Después de configurar UnityNuget, compruebe si aparecen errores de `Assembly Version Validation` en la consola de Unity. Estos errores se producen si las redirecciones de enlace para ensamblados con nombres seguros en los paquetes NuGet no se resuelven correctamente en las rutas del proyecto de Unity. Para resolver este problema, configure la validación de la versión de ensamblaje de Unity:

   a. En el editor de Unity, diríjase al menú principal y seleccione Editar, Configuración del proyecto y abra la sección Jugador.
   b. Cancele la selección de la opción Validación de la versión de ensamblaje.

Instalación del complemento

Utilice el siguiente procedimiento para instalar el complemento del SDK del servidor C# de Amazon GameLift para Unity y configurar el registro de log4net.

Para instalar el complemento

1. Con el proyecto abierto en el editor de Unity, diríjase al menú principal y seleccione Ventana, Administrador de paquetes.
2. Elija el botón + para añadir un paquete nuevo. Seleccione la opción Añadir paquete desde archivo tarball.
3. En Seleccionar paquetes en disco busque el complemento del SDK del servidor C# de Amazon GameLift para los archivos de descarga de Unity y seleccione el archivo `.tgz` del SDK del servidor de Amazon GameLift. Elija Abrir para instalar el complemento.

El SDK del servidor de Amazon GameLift utiliza el marco log4net para generar mensajes de registro. Está configurado para enviar mensajes a la terminal de una compilación de servidor de forma predeterminada, pero Unity requiere una configuración para añadir compatibilidad con el registro de archivos. Puede añadir compatibilidad a su proyecto importando la muestra proporcionada dentro del paquete del SDK del servidor de Amazon GameLift. Utilice el siguiente procedimiento para añadir la muestra y configurar log4net:

Para configurar log4net para la salida de archivos, realice el siguiente procedimiento:

1. Con el proyecto abierto en el editor de Unity, diríjase al menú principal y seleccione Ventana, Administrador de paquetes.
2. En el menú desplegable, seleccione Paquetes: en proyecto y, a continuación, seleccione el SDK del servidor de Amazon GameLift en la lista de paquetes. Se abrirán los detalles del paquete.
3. En los detalles del paquete, seleccione la opción Grupo de muestras y pulse Importar.
4. El archivo `log4net.config` y el script de `LoggingConfiguration.cs` que lo acompaña ejecutan automáticamente la configuración, que ahora está configurada en la carpeta `Assets/Samples` del proyecto.

> **Note**
>
> Si necesita mover el archivo `log4net.config` a una carpeta diferente del proyecto, también debe actualizar la ruta del archivo de configuración en el script `LoggingConfiguration.cs` con la nueva ruta. Para obtener más información, consulte el Manual de log4net sobre la configuración de log4net.

Para obtener instrucciones más detalladas y una guía de prueba, consulte el archivo `README` que se encuentra en la descarga del complemento.

Configuración una flota de Amazon GameLift Anywhere para realizar pruebas

Puede configurar su estación de trabajo de desarrollo como una flota de alojamiento de Amazon GameLift Anywhere para probar de forma iterativa su integración con Amazon GameLift. Con esta configuración, puede iniciar los procesos del servidor de juegos en su estación de trabajo, enviar solicitudes de unión o emparejamiento de jugadores a Amazon GameLift para iniciar sesiones de juego y conectar clientes a las nuevas sesiones de juego. Con su propia estación de trabajo configurada como servidor de alojamiento, podrá supervisar todos los aspectos de la integración de sus juegos con Amazon GameLift.

Para obtener instrucciones sobre cómo configurar su estación de trabajo, consulte Realización de una prueba de integración con flotas de Amazon GameLift Anywhere para completar los siguientes pasos:

1. Cree una ubicación personalizada para su estación de trabajo.
2. Cree una flota de Amazon GameLift Anywhere con su nueva ubicación personalizada. Si se realiza correctamente, esta solicitud devuelve un ID de la flota. Tome nota de ese valor, ya que lo necesitará más tarde.
3. Registre su estación de trabajo como un recurso informático en la nueva flota de Anywhere . Proporcione un nombre de procesamiento único y especifique la dirección IP de su estación de trabajo. Si se realiza correctamente, esta solicitud devuelve un punto de conexión del SDK del servicio en forma de URL de WebSocket. Tome nota de ese valor, ya que lo necesitará más tarde.
4. Genere un token de autenticación para el procesamiento de su estación de trabajo. Esta autenticación de corta duración incluye el token y una fecha de caducidad. El servidor de juegos lo usa para autenticar la comunicación con el servicio de Amazon GameLift. Guarde la autenticación en el recurso informático de su estación de trabajo para que los procesos del servidor de juegos en ejecución puedan acceder a él.

Añada el código del servidor de Amazon GameLift a su proyecto de Unity

El servidor de juegos se comunica con el servicio de Amazon GameLift para recibir instrucciones e informar sobre su estado actual. Para ello, debe añadir un código de servidor de juegos que utilice el SDK del servidor de Amazon GameLift.

El ejemplo de código proporcionado muestra los elementos básicos de integración necesarios. Utiliza `MonoBehavior` para mostrar una inicialización sencilla de un servidor de juegos con Amazon GameLift. En el ejemplo se supone que el servidor de juegos se ejecuta en una flota de Amazon GameLift Anywhere para realizar pruebas. Incluye código para lo siguiente:

- Inicializar un cliente de la API de Amazon GameLift. En el ejemplo, se utiliza la versión de `InitSDK()` con parámetros de servidor para el recurso informático y la flota de Anywhere. Utilice la URL de WebSocket, el ID de la flota, el nombre de computación (ID de host) y el token de autenticación, tal como se definió en el tema anterior Configuración una flota de Amazon GameLift Anywhere para realizar pruebas.
- Implemente funciones de devolución de llamadas para responder a las solicitudes del servicio de Amazon GameLift, incluidas `OnStartGameSession`, `OnProcessTerminate` y `onHealthCheck`.

- Llame a ProcessReady() con un puerto designado para informar al servicio Amazon GameLift cuando el proceso esté listo para alojar sesiones de juego.

El código que se presenta en este tema establece la comunicación con el servicio de Amazon GameLift. También implementa un conjunto de funciones de devolución de llamada que responden a las solicitudes. Para obtener más información sobre cada función y lo que hace el código, consulte Inicialización del proceso del servidor. Para obtener más información sobre las acciones del SDK y los tipos de datos que se utilizan en este código, consulte Referencia del SDK del servidor de Amazon GameLift para C#.

En este ejemplo se muestra cómo añadir todos los elementos necesarios, tal y como se describe en Adición de Amazon GameLift al servidor de juegos. Incluye:

Para obtener más información sobre cómo añadir funcionalidad de Amazon GameLift, consulte los siguientes temas:

- Adición de Amazon GameLift al servidor de juegos
- Referencia del SDK del servidor de Amazon GameLift para 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);
        }
    }
}
```

Recursos adicionales

Utilice los siguientes recursos para probar el servidor de juegos y ampliar la funcionalidad:

- Configure la máquina de desarrollo como una flota de Amazon GameLift Anywhere y utilícela para realizar pruebas locales. Consulte Realización de una prueba de integración de servidor personalizada.

- Cree su servidor de juegos y cargue la compilación en Amazon GameLift. Consulte Carga de una compilación del servidor de juegos personalizada en Amazon GameLift.
- Implemente la compilación de su servidor de juegos en una flota de EC2 de Amazon GameLift administrado. Consulte Creación de una nueva flota de Amazon GameLift .

Realización de una prueba de integración con flotas de Amazon GameLift Anywhere

Puede utilizar una flota de Amazon GameLift Anywhere para compilar y probar de forma iterativa la integración de sus juegos con Amazon GameLift. Configure su propio hardware como una flota de Anywhere con una conexión al servicio de Amazon GameLift y, a continuación, instale y ejecute el servidor de juegos en él. Utilice una aplicación de prueba para ejecutar escenarios como iniciar o detener sesiones de juego, realizar un seguimiento de las conexiones de los jugadores y administrar los rellenos de emparejamientos. Con una flota de Anywhere, puede actualizar la compilación del servidor de juegos según sea necesario y tener una visibilidad total de la actividad del alojamiento.

Puede utilizar flotas de Amazon GameLift Anywhere con juegos integrados en la versión 5 o superior del SDK de Amazon GameLift Server.

Temas

Desarrollo inicial

Ha desarrollado tu juego y lo está integrando con el SDK del servidor de Amazon GameLift. Para probar la integración, puede subir cada nueva iteración de la compilación del servidor de juegos a Amazon GameLift y crear una flota. Como alternativa, usar una flota de Anywhere con el portátil de desarrollo le proporciona una forma más eficiente de realizar pruebas y desarrollos iterativos.

Utilice los siguientes procedimientos para crear una flota de Anywhere e iniciar una sesión de juego en su portátil mediante la consola de Amazon GameLift o la AWS Command Line Interface (AWS CLI).

Console

1. Abra la consola de Amazon GameLift.

2. En el panel de navegación, en Alojamiento, elija Ubicaciones.
3. Elija Crear ubicación.
4. En el cuadro de diálogo Crear ubicación, realice lo siguiente:
   a. Especifique el nombre de una ubicación. De esa forma, se etiquetará la ubicación de los recursos informáticos que Amazon GameLift utiliza para ejecutar los juegos en las flotas de Anywhere. Los nombres de ubicación personalizados deben empezar por custom-.
   b. Seleccione Create (Crear).
5. Para crear una flota de Anywhere, realice el siguiente procedimiento:
   a. En el panel de navegación, en Alojamiento, elija Flotas.
   b. En la página Fleets (Flotas), elija Create fleet (Crear flota).
   c. En el paso Elegir tipo de computación, elija Anywhere y, a continuación, seleccione Siguiente.
   d. En el paso Definir detalles de flota, establezca su nueva flota. Para obtener más información, consulte Creación de una nueva flota de Amazon GameLift .
   e. En el paso Seleccionar ubicaciones, elija la ubicación personalizada creada.
   f. Complete el resto de los pasos de creación de la flota para crear su flota de Anywhere.
6. Registre el portátil como recurso informático en la flota creada. Utilice el comando `register-compute` (o la operación de la API `RegisterCompute`). Incluya el `fleet-id` creado en el paso anterior y añada un `compute-name` y la `ip-address` del portátil.

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

Ejemplo de resultados:

```
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. Inicie una sesión de depuración del servidor de juegos.

   a. Consiga el token de autorización para el portátil en la flota que ha creado. Utilice el comando `get-compute-auth-token` (o la operación de la API `GetComputeAuthToken`).

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

   Ejemplo de resultados:

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

   b. Ejecute una instancia de depuración del archivo ejecutable del servidor de juegos. Para ejecutar la instancia de depuración, el servidor de juegos debe llamar a InitSDK(). Cuando el proceso esté listo para alojar una sesión de juego, el servidor de juegos llamará a ProcessReady().

8. Cree una sesión de juego para probar su primera integración con Amazon GameLift Anywhere. Utilice el comando `create-game-session` (o la operación de la API `CreateGameSession`). Especifique la ubicación personalizada de la flota.

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

   Ejemplo de resultados:

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

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

Amazon GameLift envía un mensaje `onStartGameSession()` al proceso de servidor registrado. El mensaje contiene el objeto `GameSession` del paso anterior con las propiedades del juego, los datos de las sesiones de juego, los datos del emparejador y más información sobre la sesión de juego.

9. Añada lógica al servidor de juegos para que el proceso del servidor responda al mensaje `onStartGameSession()` con `ActivateGameSession()`. La operación envía un acuse de recibo a Amazon GameLift de que su servidor ha recibido y aceptado el mensaje de creación de la sesión de juego. Para obtener más información, consulte, Referencia del SDK del servidor de Amazon GameLift.

Su servidor de juegos ahora está ejecutando una sesión de juego para que la pruebe y la utilice en iteraciones. Para obtener información sobre cómo realizar iteraciones en el servidor de juegos, continúe en la siguiente sección.

AWS CLI

1. Cree una ubicación personalizada mediante el comando `create-location` (o la operación de la API `CreateLocation`). Una ubicación personalizada etiquetará la ubicación del hardware que Amazon GameLift utiliza para ejecutar los juegos en las flotas de Anywhere.

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

Ejemplo de resultados:

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

2. Cree una flota de Anywhere con la ubicación personalizada mediante el comando `create-fleet` (o la operación de la API `CreateFleet`). Amazon GameLift creará la flota en su región de origen y en las ubicaciones personalizadas que proporcione.

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

Ejemplo de resultados:

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

3. Registre el portátil como recurso informático en la flota creada. Utilice el comando `register-compute` (o la operación de la API `RegisterCompute`). Incluya el `fleet-id` creado en el paso anterior y añada un `compute-name` y la `ip-address` pública del portátil.

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

Ejemplo de resultados:

```
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. Inicie una sesión de depuración del servidor de juegos.

a. Consiga el token de autorización para el portátil en la flota que ha creado. Utilice el comando `get-compute-auth-token` (o la operación de la API `GetComputeAuthToken`).

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

Ejemplo de resultados:

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

b. Ejecute una instancia de depuración del archivo ejecutable del servidor de juegos. Para ejecutar la instancia de depuración, el servidor de juegos debe llamar a `InitSDK()`. Cuando el proceso esté listo para alojar una sesión de juego, el servidor de juegos llamará a `ProcessReady()`.

5. Cree una sesión de juego para probar su primera integración con Amazon GameLift Anywhere. Utilice el comando `create-game-session` (o la operación de la API `CreateGameSession`).

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

Ejemplo de resultados:

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

```
}
```

Amazon GameLift envía un mensaje `onStartGameSession()` al proceso de servidor registrado. El mensaje contiene el objeto `GameSession` del paso anterior con las propiedades del juego, los datos de las sesiones de juego, los datos del emparejador y más información sobre la sesión de juego.

6. Añada lógica al servidor de juegos para que el proceso del servidor responda al mensaje `onStartGameSession()` con `ActivateGameSession()`. La operación envía un acuse de recibo a Amazon GameLift de que su servidor ha recibido y aceptado el mensaje de creación de la sesión de juego. Para obtener más información, consulte, Referencia del SDK del servidor de Amazon GameLift.

Su servidor de juegos ahora está ejecutando una sesión de juego para que la pruebe y la utilice en iteraciones. Para obtener información sobre cómo realizar iteraciones en el servidor de juegos, continúe en la siguiente sección.

Iteración en el servidor de juegos

En este caso de uso, piense en un escenario en el que haya configurado y probado el servidor de juegos y haya detectado un error. Con Amazon GameLift Anywhere, puede iterar el código y evitar la ardua configuración que supone utilizar una flota de Amazon EC2.

1. Borre la `GameSession` existente, si es posible. Si el servidor de juegos se bloquea o no llama a `ProcessEnding()`, Amazon GameLift borrará la `GameSession` cuando el servidor de juegos deje de enviar comprobaciones de estado.
2. Realice los cambios de código en el servidor de juegos, realice la compilación y prepárese para la siguiente prueba.
3. Su flota de Anywhere anterior sigue activa y su portátil sigue registrado como recurso informático en la flota. Para volver a empezar a realizar las pruebas, cree una nueva instancia de depuración.

    a. Recupere el token de autorización para el portátil en la flota que ha creado. Utilice el comando `get-compute-auth-token` (o la operación de la API `GetComputeAuthToken`).

    ```
    aws gamelift get-compute-auth-token \
    ```

```
    --fleet-id fleet-1234 \
    --compute-name DevLaptop
```

Ejemplo de resultados:

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

b. Ejecute una instancia de depuración del archivo ejecutable del servidor de juegos. Para ejecutar la instancia de depuración, el servidor de juegos debe llamar a `InitSDK()`. Cuando el proceso esté listo para alojar una sesión de juego, el servidor de juegos llamará a `ProcessReady()`.

4. La flota tiene ahora un proceso de servidor disponible. Cree su sesión de juego y realice las próximas pruebas. Utilice el comando `create-game-session` (o la operación de la API `CreateGameSession`).

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

Amazon GameLift envía un mensaje `onStartGameSession()` al proceso de servidor registrado. El mensaje contiene el objeto `GameSession` del paso anterior con las propiedades del juego, los datos de las sesiones de juego, los datos del emparejador y más información sobre la sesión de juego.

5. Añada lógica al servidor de juegos para que el proceso del servidor responda al mensaje `onStartGameSession()` con `ActivateGameSession()`. La operación envía un acuse de recibo a Amazon GameLift de que su servidor ha recibido y aceptado el mensaje de creación de la sesión de juego. Para obtener más información, consulte, Referencia del SDK del servidor de Amazon GameLift.

Cuando termine de probar el servidor de juegos, podrá seguir utilizando Amazon GameLift para la administración de su flota y sus servidores de juegos. Para obtener más información, consulte Crea una GameLift Anywhere flota de Amazon.

Realización de una prueba de integración con Amazon GameLift Local

> **Note**
>
> Utilice este procedimiento de prueba si usa una versión del SDK del servidor de Amazon GameLift 4.x o anterior. El paquete del SDK del servidor incluye una versión compatible de Amazon GameLift Local. Si utiliza la versión 5.x del SDK del servidor, consulte Realización de una prueba de integración con flotas de Amazon GameLift Anywhere para la prueba local con una flota de Amazon GameLift Anywhere.

Utilice Amazon GameLift Local para ejecutar una versión limitada del servicio de Amazon GameLift administrado en un dispositivo local y probar cómo se integra el juego. Esta herramienta es útil al realizar el desarrollo iterativo en la integración del juego. La alternativa, que es cargar cada nueva compilación en Amazon GameLift y configurar una flota para alojar el juego, puede tardar 30 minutos o más cada vez.

Con Amazon GameLift Local, puede verificar los siguientes aspectos:

- Que el servidor de juegos se integra correctamente con el SDK del servidor y se comunica de forma adecuada con el servicio de Amazon GameLift para empezar nuevas sesiones de juego, aceptar jugadores nuevos e informar sobre el estado.
- Que el cliente de juego está integrado correctamente con el SDK de AWS para Amazon GameLift y que puede recuperar información sobre sesiones de juego existentes, iniciar sesiones de juego nuevas, conectar jugadores a juegos y conectarse a la sesión de juego.

Amazon GameLift Local es una herramienta de línea de comandos que inicia una versión autónoma del servicio de Amazon GameLift Local administrado. Amazon GameLift Local también proporciona un registro de evento de ejecución de la inicialización del proceso del servidor, comprobaciones de estado y llamadas y respuestas de la API. Amazon GameLift Local reconoce un subconjunto de las acciones del SDK de AWS para Amazon GameLift. Puede realizar llamadas desde la AWS CLI o desde el cliente de juego. Todas las acciones de la API que se ejecutan localmente lo hacen de la misma forma que en el servicio web de Amazon GameLift.

Cada proceso del servidor solo debe alojar una sesión de juego. La sesión de juego es el archivo ejecutable que se utiliza para conectarse a Amazon GameLift Local. Cuando se complete la sesión de juego, debe llamar a `GameLiftServerSDK::ProcessEndning` y salir del proceso. Al realizar

pruebas de forma local con Amazon GameLift Local, puede iniciar varios procesos del servidor. Cada proceso se conectará a Amazon GameLift Local. A continuación, podrá crear una sesión de juego para cada proceso del servidor. Cuando finalice la sesión de juego, el proceso del servidor de juegos debería cerrarse. A continuación, debe iniciar manualmente otro proceso de servidor.

Amazon GameLift local admite las siguientes API:

- CreateGameSession
- CreatePlayerSession
- CreatePlayerSessions
- DescribeGameSessions
- DescribePlayerSessions

Configuración de Amazon GameLift Local

Amazon GameLift Local se suministra en un archivo `.jar` ejecutable que se incluye en el SDK del servidor. Se puede ejecutar en Windows o Linux y utilizarse con cualquier idioma compatible con Amazon GameLift.

Antes de ejecutar Local, también debe tener instalado lo siguiente.

- Una compilación del SDK del servidor de Amazon GameLift de las versiones 3.1.5 a la 4.x.
- Java 8

Prueba de un servidor de juegos

Si solo desea probar el servidor de juegos, puede utilizar la AWS CLI para simular llamadas del cliente de juego al servicio de Amazon GameLift Local. De este modo, se verifica que el servidor de juegos se comporta según lo esperado:

- El servidor de juegos se inicia correctamente e inicializa el SDK del servidor de Amazon GameLift.
- Como parte del proceso de lanzamiento, el servidor de juegos notifica a Amazon GameLift que el servidor está listo para alojar sesiones de juego.
- El servidor de juegos envía estados a Amazon GameLift cada minuto mientras se ejecuta.
- El servidor de juegos responde a las solicitudes para iniciar una sesión de juego nueva.

1. Inicie Amazon GameLift Local.

   Abra una ventana de símbolo del sistema, vaya al directorio que contiene el archivo *GameLiftLocal.jar* y ejecútelo. De forma predeterminada, Local atiende a las solicitudes de clientes de juego en el puerto 8080. Para especificar un número de puerto diferente, utilice el parámetro `-p`, tal y como se muestra en el ejemplo siguiente:

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

   En cuanto Local arranque, verá los registros que indican que se iniciaron dos servidores locales, uno que atiende al servidor de juegos y otro al cliente de juego o a la AWS CLI. Los registros continúan informando sobre la actividad de los dos servidores locales, incluida la comunicación a y desde los componentes de juego.

2. Inicie el servidor de juegos.

   Arranque el servidor de juegos integrado en Amazon GameLift localmente. No es necesario cambiar el punto de enlace del servidor de juegos.

   En la ventana de símbolo del sistema de Local, los mensajes de registro indican que el servidor de juegos se ha conectado al servicio de Amazon GameLift Local. Esto significa que el servidor de juegos ha inicializado correctamente el SDK del servidor de Amazon GameLift (con `InitSDK()`). Ha llamado a `ProcessReady()` con las rutas de registro que se muestran y, en caso de éxito, está listo para alojar una sesión de juego. Mientras se ejecuta el servidor de juegos, Amazon GameLift registra cada informe de estado del servidor de juegos. El siguiente ejemplo de mensajes de registro muestra un servidor de juegos integrado correctamente:

   ```
   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
```

A continuación se presentan posibles mensajes de error y advertencia:

- Error: «ProcessReady no encontró un proceso con pID: *<ID de proceso>*. ¿Se ha invocado InitSDK()?»
- Advertencia: «El estado del proceso ya existe para el proceso con pID: *<ID de proceso>*. ¿Se ha invocado ProcessReady(...) más de una vez?»

3. Inicie la AWS CLI.

   En cuanto el servidor de juegos llame a `ProcessReady()` correctamente, podrá comenzar a realizar llamadas de cliente. Abra una ventana de símbolo del sistema y empiece con la herramienta AWS CLI. De forma predeterminada, la AWS CLI utiliza el punto de conexión de servicio web de Amazon GameLift. Debe anularlo con el punto de enlace de Local en cada solicitud que emplee el parámetro `--endpoint-url`, tal y como se muestra en el siguiente ejemplo.

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

   En la ventana de símbolo del sistema de la AWS CLI, los comandos de `AWS gamelift` generan respuestas similares a las documentadas en la Referencia de comandos de la AWS CLI.

4. Cree una sesión de juego.

   Con la AWS CLI, envíe una solicitud CreateGameSession(). La solicitud debería seguir la sintaxis esperada. Para Local, el parámetro `FleetId` puede adaptarse a cualquier cadena válida (`^fleet-\S+`).

   ```
   AWS gamelift create-game-session --endpoint-url http://localhost:9080 --maximum-
   player-session-count 2 --fleet-id
       fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d
   ```

   En la ventana de símbolo del sistema de Local, los mensajes de registro indican que Amazon GameLift Local ha enviado una devolución de la llamada `onStartGameSession` al servidor de

juegos. Si se crea una sesión de juego correctamente, el servidor de juegos responderá llamado a `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"
```

En la ventana de la AWS CLI, Amazon GameLift responde con un objeto de sesión de juego que incluye un ID de sesión de juego. Observe que el estado de la nueva sesión de juego es Activating. El estado cambia a Active cuando el servidor de juegos invoca a ActivateGameSession. Si desea ver el estado modificado, utilice la AWS CLI para llamar a `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
    }
}
```

Prueba de un servidor y un cliente de juegos

Para comprobar la integración total del juego, incluidos los jugadores conectados a los juegos, puede ejecutar tanto el servidor como el cliente de juego localmente. De este modo podrá realizar llamadas programáticas desde su cliente de juego a Amazon GameLift Local. Puede verificar las siguientes acciones:

- El cliente de juego está realizando correctamente solicitudes del SDK de AWS al servicio de Amazon GameLift Local, incluida la creación de sesiones de juego, la recuperación de información sobre sesiones de juego existentes y la creación de sesiones de jugador.
- El servidor de juegos está validando jugadores correctamente al intentar conectarse a una sesión de juego. Para validar los jugadores, el servidor de juego puede recuperar datos de los jugadores (si están implementados).
- El servidor de juegos informa sobre la pérdida de la conexión cuando un jugador abandona el juego.
- El servidor de juegos informa sobre la finalización de una sesión de juego.

1. Inicie Amazon GameLift Local.

   Abra una ventana de símbolo del sistema, vaya al directorio que contiene el archivo *`GameLiftLocal.jar`* y ejecútelo. De forma predeterminada, Local atiende a las solicitudes de clientes de juego en el puerto 8080. Para especificar un número de puerto diferente, utilice el parámetro `-p`, tal y como se muestra en el ejemplo siguiente.

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

   En cuanto Local arranque, verá los registros que indican que se iniciaron dos servidores locales, uno que atiende al servidor de juegos y otro al cliente de juego o a la AWS CLI.
2. Inicie el servidor de juegos.

   Arranque el servidor de juegos integrado en Amazon GameLift localmente. Consulte Prueba de un servidor de juegos para obtener más información acerca de los registros de mensajes.
3. Configure su cliente de juego para Local e inícielo.

   Para utilizar el cliente de juego con el servicio de Amazon GameLift Local, debe realizar los siguientes cambios en la configuración del cliente de juego, tal y como se describe en Configurar Amazon GameLift en un servicio de back-end:

- Modifique el objeto `ClientConfiguration` para que apunte hacia el punto de enlace de Local, por ejemplo `http://localhost:9080`.
- Defina un valor de ID de la flota de destino. Para Local, no se requiere un ID de flota real; defina la flota de destino con cualquier cadena válida (`^fleet-\S+`), como `fleet-1a2b3c4d-5e6f-7a8b-9c0d-1e2f3a4b5c6d`.
- Defina las credenciales de AWS. Para Local, no requiere credenciales de AWS reales, de modo que puede definir la clave de acceso y la clave secreta para cualquier cadena.

En la ventana de símbolo del sistema de Local, después de iniciar el cliente de juego, los mensajes de registro deberían indicar que ha inicializado `GameLiftClient` y que se comunica correctamente con el servicio de Amazon GameLift.

4. Pruebe las llamadas del cliente de juego al servicio de Amazon GameLift.

   Compruebe que el cliente de juego realiza correctamente alguna o todas las llamadas a la API siguientes:

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

   En la ventana de símbolo del sistema de Local, solo las llamadas a `CreateGameSession()` resultan en mensajes de registro. Los mensajes de registro muestran el momento en el que Amazon GameLift Local solicita al servidor de juegos que inicie una sesión de juego (devolución de llamada `onStartGameSession`) y obtiene una `ActivateGameSession` correcta cuando el servidor de juegos lo invoca. En la ventana de la AWS CLI, todas las llamadas a la API generan respuestas o mensajes de error tal y como se documenta.

5. Compruebe que el servidor de juegos valida las conexiones de jugadores nuevas.

   Después de crear una sesión de juego y una sesión de jugador, establezca una conexión directa con la sesión de juego.

   En la ventana de símbolo del sistema de Local, los mensajes de registro deberían mostrar que el servidor de juegos ha enviado una solicitud `AcceptPlayerSession()` para validar la conexión

de jugadores nueva. Si utiliza la AWS CLI para llamar a `DescribePlayerSessions()`, el estado de la sesión de jugador debería cambiar de Reserved a Active.

6. Verifique que el servidor de juegos informe sobre el estado del juego y del jugador al servicio de Amazon GameLift.

   Para que Amazon GameLift administre la demanda de los jugadores y notifique correctamente las métricas, el servidor de juegos debe informar sobre varios estados a Amazon GameLift. Compruebe que Local registra los eventos relacionados con las acciones siguientes. Es posible que también desee utilizar la AWS CLI para realizar un seguimiento de los cambios de estado.

   - El jugador se desconecta de una sesión de juego: los mensajes de registro de Amazon GameLift Local deberían mostrar que el servidor de juegos llama a `RemovePlayerSession()`. Una llamada de la AWS CLI a `DescribePlayerSessions()` debería mostrar un cambio de estado de `Active` a `Completed`. También puede llamar a `DescribeGameSessions()` para comprobar que el recuento de jugadores actual de la sesión de juego resulta en un jugador menos.
   - La sesión de juego finaliza: los mensajes de registro de Amazon GameLift Local deberían mostrar que el servidor de juegos llama a `TerminateGameSession()`.

     > **Note**
     >
     > La guía anterior consistía en llamar a `TerminateGameSession()` al finalizar una sesión de juego. Este método ha quedado obsoleto en la versión 4.0.1 del SDK de Amazon GameLift Server. Consulte Finalización de una sesión de juego.

   - El proceso del servidor ha finalizado: los mensajes de registro de Amazon GameLift Local deberían mostrar que el servidor de juegos llama a `ProcessEnding()`. Una llamada de la AWS CLI a `DescribeGameSessions()` debería mostrar un cambio de estado de `Active` a `Terminated` (o `Terminating`).

Variaciones de Local

Al utilizar Amazon GameLift Local, tenga en cuenta los siguientes aspectos:

- A diferencia del servicio web de Amazon GameLift, Local no realiza el seguimiento del estado del servidor ni inicia la devolución de la llamada `onProcessTerminate`. Local simplemente deja de registrar los informes de estado del servidor de juegos.

- Para realizar llamadas al SDK de AWS, los ID de la flota no están validados y puede tratarse de cualquier valor de cadena que cumpla los requisitos de parámetro (`^fleet-\S+`).
- Los ID de sesión de juego creados con Local tienen una estructura diferente. Incluyen la cadena `local`, tal y como se muestra aquí:

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

Integración de juegos con Servidores en tiempo real de Amazon GameLift

En este tema se proporciona una descripción general de Amazon GameLift administrado con la solución de Realtime Servers. La descripción general explica cuándo esta solución es adecuada para el juego y cómo Realtime Servers admite los juegos multijugador.

Si desea ver una hoja de ruta completa para poner a punto su juego y ejecutarlo, consulte Hoja de ruta de alojamiento administrado de Amazon GameLift.

> **Tip**
>
> Para probar el alojamiento del servidor de juegos de Amazon GameLift, consulte Introducción a Amazon GameLift.

¿Qué son los servidores de Realtime?

Los servidores de Realtime son servidores de juegos ligeros y listos para utilizarse proporcionados por Amazon GameLift para que los utilice con sus juegos multijugador. Los servidores de Realtime eliminan el proceso de desarrollo, prueba e implementación de un servidor de juegos personalizado. Esta solución puede ayudar a minimizar el tiempo y el esfuerzo necesarios para completar el juego.

Características principales

- Pila de red completa para la interacción del cliente y servidor de juegos
- Funcionalidad principal del servidor de juegos
- Lógica del servidor personalizable

- Actualizaciones en directo de la lógica del servidor y las configuraciones de Realtime.
- Emparejamiento de FlexMatch
- Control flexible de los recursos de alojamiento

Cree una flota y proporcione un script de configuración para configurar los servidores para configurar servidores de Realtime. Para obtener más información sobre cómo crear servidores de Realtime y cómo preparar el cliente de juegos, consulte Preparación del servidor de Realtime.

Cómo Realtime Servers administra las sesiones de juego

Tiene la opción de añadir lógica personalizada para la administración de sesiones de juego si la compila en el script de Realtime. Puede escribir código para obtener acceso a objetos específicos del servidor, añadir lógica basada en eventos mediante devoluciones de llamada o añadir lógica basada en escenarios que no sean eventos.

Cómo interactúan los clientes y servidores de Realtime

Durante una sesión de juego, los clientes de juegos interactúan enviando mensajes al servidor de Realtime a través de un servicio de backend. Después, el servicio de backend transmite los mensajes entre los clientes del juego para intercambiar la actividad, el estado del juego y los datos relevantes del juego.

Además, añada la lógica de juego al script de Realtime para personalizar la forma en que los clientes y los servidores interactúan. Con la lógica de juego personalizada, un servidor de Realtime podría implementar devoluciones de llamada para iniciar respuestas basadas en eventos.

Protocolo de comunicación

Los servidores de Realtime y los clientes de juegos conectados se comunican a través de dos canales: una conexión TCP para ofrecer una entrega de confianza y un canal UDP para proporcionar una entrega rápida. Al crear mensajes, los clientes de juego eligen qué protocolo utilizar en función de la naturaleza de los mensajes. La entrega de mensajes se configura como UDP de forma predeterminada. Si no hay un canal UDP disponible, Amazon GameLift envía los mensajes mediante TCP como alternativa.

Contenido de los mensajes

El contenido del mensaje consta de dos elementos: un código de operación obligatorio (opCode) y una carga opcional. El opCode de un mensaje identifica una actividad de jugador o un evento

de juego en particular, mientras que la carga facilita datos adicionales relacionados con el código de operación. Ambos elementos están definidos por el desarrollador. El cliente de juegos actúa en función de los opCodes en los mensajes que recibe.

Grupos de jugadores

Realtime Servers proporciona funcionalidad para administrar grupos de jugadores. De forma predeterminada, Amazon GameLift ubica todos los jugadores que se conectan a un juego en un grupo "todos los jugadores". Además, los desarrolladores pueden definir otros grupos para sus juegos y los jugadores pueden ser miembros de varios grupos de forma simultánea. Los miembros del grupo pueden enviar mensajes y compartir los datos del juego con todos los jugadores del grupo. Un posible uso de los grupos consiste en configurar equipos de jugadores y administrar la comunicación de los equipos.

Realtime Servers con certificados TLS

Con Realtime Servers, la autenticación del servidor y el cifrado de los paquetes de datos están integrados en el servicio. Estas características de seguridad se habilitan al activar la generación de certificados TLS. Cuando el cliente del juego intenta conectarse a un servidor de Realtime, el servidor responde automáticamente con el certificado TLS, que el cliente valida. Amazon GameLift administra el cifrado mediante la comunicación TLS para TCP (Websockets) y DTLS para el tráfico UDP.

Personalización de un servidor de Realtime

Un servidor de Realtime funciona como un servidor de retransmisión sin estado. El servidor de Realtime retransmite paquetes de datos del juego y los mensajes entre los clientes del juego conectados al juego. Sin embargo, el servidor de Realtime no evalúa los mensajes, no procesa datos ni ejecuta ninguna lógica del juego. Si se utiliza de esta forma, cada cliente del juego mantiene su propia vista del estado del juego y facilita actualizaciones a otros jugadores mediante el servidor de retransmisión. Cada cliente de juego es responsable de incorporar estas actualizaciones y conciliar su propio estado del juego.

Puede personalizar los servidores añadiéndolos a la funcionalidad de script de Realtime. Con la lógica del juego, por ejemplo, puede compilar un juego con estado con una vista autorizada por el servidor del estado del juego.

Amazon GameLift define un conjunto de devoluciones de llamada del lado del servidor para scripts de Realtime. Implemente estas devoluciones de llamada para añadir a su servidor la funcionalidad basada en eventos. Por ejemplo, puede hacer lo siguiente:

- Autenticar a un jugador cuando un cliente de juego intenta conectarse con el servidor.
- Valide si un jugador podrá unirse a un grupo cuando se solicite.
- Establezca cuándo entregar mensajes de un jugador determinado a otro jugador de destino, o realizar un procesamiento adicional en la respuesta.
- Informe a todos los jugadores cuando un jugador abandone un grupo o se desconecte del servidor.
- Visualice el contenido de los objetos de la sesión de juego o los objetos de mensajes, y utilice los datos.

Implementación y actualización de Realtime Servers

Una ventaja clave de Realtime Servers es la capacidad de actualizar los scripts en cualquier momento. Al actualizar un script, Amazon GameLift distribuirá la nueva versión a los recursos de alojamiento en cuestión de minutos. Una vez que Amazon GameLift implemente el nuevo script, todas las sesiones de juego nuevas que se hayan creado después de ese momento utilizarán la nueva versión del script. (Las sesiones de juego existentes seguirán utilizando la versión original).

Comience la integración del juego con Realtime Servers de las siguientes formas:

- Integración de un cliente de juegos para Realtime Servers
- Creación de un script de Realtime

Integración de un cliente de juegos para Realtime Servers

En este tema se describe cómo preparar su cliente de juegos para poder unirse y participar en sesiones de juego alojadas en Amazon GameLift.

Hay dos conjuntos de tareas necesarias para preparar su cliente de juego:

- Configure su cliente de juego para adquirir información sobre juegos existentes, solicitar emparejamientos, iniciar sesiones de juego nuevas y reservar ranuras de sesiones de juego para un jugador.
- Habilite su cliente de juegos para unirse a una sesión de juego alojada en un servidor de Realtime e intercambiar mensajes.

Búsqueda o creación de sesiones de juego y sesiones de jugador

Configure su cliente de juegos para encontrar o iniciar sesiones de juego, solicitar emparejamientos de FlexMatch y reservar espacio para los jugadores en un juego mediante la creación de sesiones de jugador. Como práctica recomendada, cree un servicio de backend y utilícelo para realizar las solicitudes directas al servicio de Amazon GameLift cuando se desencadenen debido a una acción cliente de juegos. El servicio de backend transmite a continuación respuestas relevantes de vuelta al cliente de juegos.

1. Añada el SDK de AWS a su cliente de juegos, inicialice un cliente de Amazon GameLift y configúrelo para utilizar los recursos de alojamiento en sus flotas y colas. El SDK de AWS está disponible en varios idiomas; consulte los SDK de Amazon GameLift Para servicios de cliente personalizados, realice el siguiente procedimiento:.
2. Añada la funcionalidad de GameLift a su servicio de backend. Para obtener instrucciones más detalladas, consulte Añade Amazon GameLift a tu cliente de juegos y Adición del emparejamiento de FlexMatch. La práctica recomendada consiste en utilizar las ubicaciones de sesiones de juego para crear nuevas sesiones de juego. Este método le permite aprovechar al máximo la capacidad de GameLift de colocar de manera rápida e inteligente nuevas sesiones de juego, además de utilizar los datos de latencia del jugador para minimizar el retardo del juego. Como mínimo, el servicio de backend debe ser capaz de solicitar nuevas sesiones de juego y de gestionar los datos de la sesión como respuesta. Es posible que también desee añadir funcionalidades para buscar y obtener información sobre sesiones de juego existentes y solicitar sesiones de jugador, lo que, de hecho, reserva un espacio de jugador en una sesión de juego existente.
3. Devolver la información de conexión al cliente del juego. El servicio de backend recibe objetos de sesión de juego y sesión de jugador en respuesta a las solicitudes al servicio de Amazon GameLift. Estos objetos contienen información, especialmente detalles de conexión (dirección IP y puerto) e ID de sesión de jugador, que el cliente de juego tiene que conectar a la sesión de juego que se ejecuta en un servidor Realtime.

Conexión a juegos en Realtime Servers

Permita que su cliente de juegos se conecte directamente a una sesión de juego alojada en un servidor de Realtime e intercambie mensajes con el servidor y otros jugadores.

1. Obtenga el SDK de cliente de Realtime, compílelo y añádalo a su proyecto de cliente de juegos. Consulte el archivo README para obtener más información sobre los requisitos del SDK e instrucciones sobre cómo crear las bibliotecas de cliente.

2. Llame a Client() con una configuración de cliente que especifique el tipo de conexión cliente/servidor que se va a utilizar.

> **Note**
>
> Si se conecta a un servidor Realtime que se ejecuta en una flota protegida con un certificado TLS, debe especificar un tipo de conexión segura.

3. Añada la siguiente funcionalidad a su cliente de juego. Consulte Referencia de la API de cliente de Realtime Servers (C#) para obtener más información.
   - Conectarse a y desconectarse de un juego
     - Connect()
     - Disconnect()
   - Enviar mensajes a los destinatarios
     - SendMessage()
   - Recibir y procesar mensajes
     - OnDataReceived()
   - Unirse a grupos y abandonar grupos de jugadores
     - JoinGroup()
     - RequestGroupMembership()
     - LeaveGroup()
4. Configurar controladores de eventos para devoluciones de llamadas del cliente según sea necesario. Consulte Referencia de la API de cliente de Realtime Servers (C#) de : Devoluciones de llamadas asíncronas.

Cuando se trabaja con flotas de Realtime que tienen habilitada la generación de certificados TLS, el servidor se autentica automáticamente con el certificado TLS. El tráfico TCP y UDP se cifra en tránsito para proporcionar seguridad en la capa de transporte. El tráfico TCP se cifra con TLS 1.2 y el tráfico UDP se cifra con DTLS 1.2.

Ejemplos de cliente de juegos

Cliente básico de Realtime (C#)

Este ejemplo muestra una integración del cliente de juegos básico con el SDK de cliente de Realtime (C#). Tal y como se muestra, en el ejemplo se inicializa un objeto de cliente de Realtime, se configuran controladores de eventos, se implementan las devoluciones de llamada del lado del cliente, se conecta a un servidor de Realtime, se envía un mensaje y se desconecta.

```
using System;
using System.Text;
using Aws.GameLift.Realtime;
using Aws.GameLift.Realtime.Event;
using Aws.GameLift.Realtime.Types;

namespace Example
{
    /**
     * An example client that wraps the 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);
        }
    }
}
```

Creación de un script de Realtime

Para utilizar Realtime Servers para el juego, debe proporcionar un script (en forma de código JavaScript) para configurar y, de forma opcional, personalizar una flota de Realtime Servers. En este tema se explican los pasos principales para crear un script de Realtime. Una vez que el script esté listo, cárguelo en el servicio de Amazon GameLift y utilícelo para crear una flota (consulte Carga de un script de Realtime Servers en Amazon GameLift).

Para preparar un script para su uso con Realtime Servers, añada la siguiente funcionalidad en su script de Realtime.

Administración del ciclo de vida de la sesión de juego (obligatorio)

Como mínimo, un script de Realtime debe incluir la función `Init()`, que prepara el servidor de Realtime para iniciar una sesión de juego. También es muy recomendable que proporcione además una forma de terminar las sesiones de juego, para garantizar que las nuevas sesiones de juego se puedan seguir iniciando en la flota.

La función de devolución de llamada `Init()`, cuando se llama, se transfiere un objeto de sesión de Realtime, que contiene una interfaz para el servidor de Realtime. Consulte Interfaz de Realtime Servers para obtener más detalles sobre esta interfaz.

Para finalizar una sesión de juego con fluidez, el script también debe llamar a la función `session.processEnding` del servidor de Realtime. Esto requiere algún mecanismo para determinar cuándo finalizar una sesión. El código de ejemplo de script ilustra un mecanismo sencillo que comprueba las conexiones de jugadores y desencadena la finalización de la sesión de juego cuando no se ha conectado ningún jugador a la sesión durante un período de tiempo especificado.

Los servidores de Realtime Servers con la configuración más básica (inicialización y terminación de procesos del servidor) actúan básicamente como servidores de retransmisión sin estado. El servidor de Realtime transmite los mensajes y datos del juego entre los clientes que se conectan al juego, pero no realiza ninguna acción independiente para procesar los datos o ejecutar la lógica. Si lo desea, puede añadir lógica del juego que se desencadene a partir de eventos del juego u otros mecanismos, según sea necesario para el juego.

Adición de lógica de juego del lado del servidor (opcional)

Si lo desea, puede añadir la lógica del juego a su script de Realtime. Por ejemplo, es posible que realice alguna de las siguientes acciones. El código de ejemplo de script proporciona una ilustración. Consulte Referencia de scripts de Servidores en tiempo real de Amazon GameLift.

- Añadir lógica basada en eventos. Implemente las funciones de devolución de llamada para responder a eventos de cliente-servidor. Consulte Devoluciones de llamadas de script para Realtime Servers para obtener una lista completa de devoluciones de llamada.
- Desencadenar la lógica enviando mensajes al servidor. Cree un conjunto de códigos de operación especiales para los mensajes enviados desde clientes de juego al servidor y añada funciones para gestionar la recepción. Utilice la devolución de llamada `onMessage` y analice el contenido de los mensajes mediante la interfaz `gameMessage` (consulte gameMessage.opcode).
- Habilite la lógica del juego para acceder a otros recursos de AWS. Para obtener más información, consulte Comunicación con otros recursos de AWS de sus flotas.

- Permite que la lógica del juego acceda a la información de la flota para la instancia en la que se esté ejecutando. Para obtener más información, consulte Obtención de datos de la flota para una instancia de Amazon GameLift.

Ejemplo del script de Realtime Servers

Este ejemplo muestra un script básico necesario para implementar Realtime Servers además de alguna lógica personalizada. Contiene la función `Init()` requerida y utiliza un mecanismo de temporizador para desencadenar la finalización de sesiones de juego basada en la duración de tiempo sin conexiones de jugadores. También incluye algunos enlaces para lógica personalizada, incluidas algunas implementaciones de devolución de llamada.

```
// 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
};
```

Integración de juegos con el GameLift complemento Amazon para Unity

Los temas de esta sección describen el GameLift complemento de Amazon para Unity y cómo usarlo para preparar tu proyecto de juego multijugador para alojarlo en Amazon GameLift. Trabaje completamente en su entorno de desarrollo de Unity con los flujos de trabajo guiados del complemento para completar los requisitos básicos de alojamiento con Amazon GameLift.

Amazon GameLift es un servicio totalmente gestionado que permite a los desarrolladores de juegos gestionar y escalar servidores de juegos dedicados para juegos multijugador basados en sesiones. Para obtener más información sobre el GameLift alojamiento de Amazon, consulteCómo funciona Amazon GameLift.

- Guía GameLift del complemento de Amazon para Unity para el SDK de servidor 5.x, versión 2.0.0, funciona con el SDK 5.x del servidor y es compatible con Amazon. GameLift Anywhere
- Guía GameLift del complemento de Amazon para Unity para el SDK 4.x del servidor, versión 1.0.0, funciona con el SDK de servidor 4.x o versiones anteriores. Esta versión usa Amazon GameLift Local para las pruebas de integración.

Guía GameLift del complemento de Amazon para Unity para el SDK de servidor 5.x

Amazon GameLift proporciona herramientas para preparar tus servidores de juegos multijugador para que funcionen con Amazon GameLift. El GameLift complemento Amazon para Unity facilita la integración de Amazon GameLift en tus proyectos de juegos de Unity, prueba tu integración con Amazon GameLift Anywhere y despliega GameLift los recursos de Amazon para el alojamiento en la nube.

Este complemento usa AWS CloudFormation plantillas para implementar soluciones de alojamiento para escenarios de juego comunes. Usa estas soluciones tal y como se proporcionan o personalízalas según sea necesario para tus juegos.

Temas

Acerca del complemento

El complemento para Unity proporciona una experiencia de inicio simplificada para integrar y alojar tus juegos multijugador de Unity con Amazon GameLift. Puedes aprovechar la funcionalidad del plugin y los componentes prediseñados para poner en marcha tus juegos rápidamente.

El plugin añade herramientas y funcionalidades al editor de Unity. Usa los flujos de trabajo guiados para GameLift integrar Amazon en tu proyecto de juego, pruébalo localmente y, a continuación, implementa el servidor del juego en el alojamiento GameLift en la nube de Amazon.

Utilice las soluciones de alojamiento prediseñadas del complemento para implementar su juego. Configure una flota de Amazon GameLift Anywhere con su estación de trabajo local como host. Para el alojamiento en la nube, elija entre dos escenarios de implementación comunes que equilibren la latencia de los jugadores, la disponibilidad de las sesiones de juego y el costo de diferentes maneras. Uno de los escenarios incluye un sencillo sistema de FlexMatch emparejamiento y un conjunto de reglas. Utilice estos escenarios para implementar una solución de alojamiento básica lista para la producción y, a continuación, optimícela y personalice según sea necesario.

El complemento incluye los siguientes componentes:

- Módulos de complementos para el editor de Unity. Cuando se instala el complemento, un nuevo elemento del menú principal te da acceso a las GameLift funciones de Amazon.
- Bibliotecas de C# para la API de GameLift servicios de Amazon con funcionalidad del lado del cliente.
- Bibliotecas de C# para el SDK GameLift del servidor Amazon (versión 5.x).
- Prueba el contenido del juego, incluidos los activos y las escenas, para que puedas probar Amazon GameLift incluso si no tienes un juego multijugador listo para compilar.
- Configuraciones de soluciones, proporcionadas en forma de AWS CloudFormation plantillas, que el plugin utiliza cuando despliega tu servidor de juegos en la nube como alojamiento.

Flujo de trabajo del complemento

Los siguientes pasos describen un enfoque típico para integrar e implementar un proyecto de juego con el GameLift complemento de Amazon para Unity. Para completar estos pasos, debes trabajar en el editor de Unity y en el código del juego.

1. Crea un perfil de usuario que se vincule a tu AWS cuenta y proporcione las credenciales de acceso de un usuario de cuenta válido con permisos para usar Amazon GameLift.
2. Añade el código de servidor a tu proyecto de juego para establecer la comunicación entre un servidor de juegos en ejecución y el GameLift servicio with Amazon.
3. Añade un código de cliente a tu proyecto de juego que permita a los clientes del juego enviar solicitudes GameLift a Amazon para iniciar una sesión de juego o unirse a ella y, a continuación, conectarse al servidor del juego.
4. Utilice el flujo de trabajo de Anywhere para configurar su estación de trabajo local como alojamiento de Anywhere para su servidor de juegos. Abre el servidor y el cliente del juego de forma local, conéctate a una sesión de juego y prueba la integración.

5. Utilice el flujo de trabajo de alojamiento de EC2 para cargar su servidor de juegos integrado e implementar una solución de alojamiento en la nube. Cuando el servidor de juegos esté listo, inicie el cliente de juego de forma local, conéctese a una sesión de juego, inicie sesión y juegue.

Cuando utilice el complemento, creará y utilizará recursos de AWS. Esas acciones pueden conllevar gastos en la cuenta de AWS que use. Si es la primera vez que lo AWS usas, es posible que las acciones estén incluidas en la capa AWS gratuita.

Instala el complemento para Unity

En esta sección se describe cómo agregar el complemento a un proyecto de Unity. Una vez instalado el complemento, la funcionalidad del complemento estará disponible cuando tengas el proyecto abierto en el editor de Unity.

Antes de comenzar

Esto es lo que necesitas para usar el GameLift complemento Amazon para Unity:

- Unity para Windows 2022 LTS o Unity para macOS
- Descarga GameLift del plugin de Amazon para Unity. [Sitio de descargas] La descarga incluye dos paquetes:
  - Complemento GameLift independiente de Amazon para Unity
  - SDK de servidor Amazon GameLift C# para Unity
- Microsoft Visual Studio 2019 o una versión posterior.
- Un proyecto de juego multijugador con código de juego en C#.
- El registro controlado por terceros. UnityNuGet Esta herramienta administra archivos DLL de terceros. Para obtener más información, consulta el repositorio de UnityNuGetGithub.

Adición del complemento al proyecto de juego

Completa las siguientes tareas trabajando en el editor de Unity y en los archivos del proyecto del juego.

Paso 1: agrégalo UnityNuGet a tu proyecto de juego

Si no has UnityNuGet configurado tu proyecto de juego, sigue los siguientes pasos para instalar la herramienta mediante el administrador de paquetes de Unity. Como alternativa, puede usar la

NuGet CLI para descargar manualmente las DLL. Para obtener más información, consulta el SDK de servidor Amazon GameLift C# para UnityREADME.

1. Con tu proyecto abierto en el editor de Unity, ve al menú principal y selecciona Editar, Configuración del proyecto. Entre las opciones, elija la sección Administrador de paquetes y abra el grupo Registros con ámbito.
2. Selecciona el botón + e introduce los siguientes valores para el registro UnityNuGet abarcado:

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

Para los usuarios de la versión Unity 2021:

Tras la configuración UnityNuGet, comprueba si aparecen `Assembly Version Validation` errores en la consola de Unity. Estos errores se producen si los redireccionamientos de enlace de los ensamblajes con nombres fuertes de los NuGet paquetes no se resuelven correctamente en las rutas del proyecto de Unity. Para resolver este problema, configure la validación de la versión de ensamblaje de Unity:

1. En el editor de Unity, ve al menú principal y selecciona Editar, Configuración del proyecto y abre la sección Reproductores.
2. Cancele la selección de la opción Validación de la versión de ensamblaje.

Paso 2: Agrega el plugin y los paquetes SDK del servidor C#

1. Descomprime el GameLift plugin de Amazon para descargar Unity, que contiene ambos paquetes.
2. Con tu proyecto abierto en el Editor de Unity, ve al menú principal y selecciona Window, Package Manager.
3. Elija el botón + para añadir un paquete nuevo. Seleccione la opción Añadir paquete desde archivo tarball.
4. En Seleccionar paquetes en disco, busca el complemento Amazon GameLift C# Server SDK para archivos de descarga de Unity y selecciona el `com.amazonaws.gameliftserver.sdk-<version>.tgz` archivo. Elija Abrir para instalar el complemento.

5. En Seleccionar paquetes en disco, busca el complemento GameLift independiente de Amazon para descargar archivos de Unity y selecciona el archivo`com.amazonaws.gamelift-<version>.tgz`. Elija Abrir para instalar el complemento.
6. Verifica que el complemento independiente se haya agregado a tu proyecto. Regresa a la ventana del editor de Unity. Consulta el menú principal para ver el nuevo botón de GameLift menú de Amazon.

Paso 3: Importa el juego de muestra (opcional)

El complemento para Unity incluye un conjunto de ejemplos de activos de juego, incluidas escenas, que puedes añadir a tu proyecto de juego. La importación del juego de muestra te ofrece una forma rápida de probar, crear e implementar un juego multijugador sencillo con Amazon GameLift. El juego de muestra ya está totalmente integrado con GameLift los SDK de Amazon, por lo que puedes saltarte las tareas de integración y completar las demás tareas del flujo de trabajo.

Al usar el juego de muestra, puedes configurar y unirte a una flota de Amazon GameLift Anywhere alojada localmente en solo unos minutos. Puedes implementar el juego en Amazon GameLift y unirte a un juego en vivo alojado en la nube en menos de una hora.

Para importar el juego de muestra:

1. Con tu proyecto de juego abierto en el Editor de Unity, ve al GameLift menú de Amazon y selecciona Sample Game, Import Sample Game.
2. Una vez importados los archivos, vuelve al GameLift menú de Amazon y selecciona Sample Game, Initialize Settings. Este paso configura tu proyecto para crear el cliente y el servidor del juego.

Cuando se complete la instalación, verás dos escenas nuevas añadidas a tu proyecto de juego. También verás algunos recursos adicionales del proyecto, incluido un GameLiftClientSettingsactivo.

Para obtener más información sobre la interfaz de usuario y la jugabilidad del ejemplo, consulta el archivo readme del juego de muestra.

Configuración de un perfil de usuario de AWS

Después de instalar el complemento, configure un perfil y vincúlelo a un usuario de la cuenta de AWS válido. Puede mantener varios perfiles, pero solo puede tener un perfil activo a la vez. Siempre que trabaje con el complemento, seleccione el perfil que quiera utilizar.

El mantenimiento de varios perfiles le ofrece la posibilidad de cambiar entre diferentes escenarios de alojamiento. Por ejemplo, puede configurar perfiles con las mismas credenciales de AWS, pero con distintas regiones de AWS. También puede configurar perfiles con diferentes cuentas de AWS o con distintos conjuntos de usuarios o permisos.

> **Note**
>
> Si ha instalado la AWS CLI en su estación de trabajo y ya tiene un perfil configurado, el GameLift complemento Amazon puede detectarlo y lo incluirá como un perfil existente. El complemento selecciona automáticamente cualquier perfil nombrado como `[default]`. Puede utilizar un perfil existente o crear uno nuevo.

Para configurar su perfil AWS

1. En el menú principal del editor de Unity, elige Amazon GameLift y selecciona Establecer perfiles de AWS cuenta. Esta acción abre la ventana del plugin. Abre la página Perfiles AWS de usuario.
2. Si el complemento detecta un perfil existente, no se le pedirá que cree uno. Selecciona un perfil existente o elige Añadir otro perfil para crear uno nuevo.
3. Si el complemento no detecta un perfil existente, le solicitará que cree uno. Puede crear un perfil nuevo mediante una cuenta de AWS nueva o existente.

   > **Note**
   >
   > Debe usar la consola de administración de AWS para crear una cuenta de AWS nueva y crear o actualizar un usuario con el conjunto de permisos adecuado.

   Al configurar un perfil, deberá proporcionar la siguiente información:

   - Una cuenta de AWS. Si necesita crear una cuenta de AWS nueva, siga las instrucciones para crearla. Consulte Creación de una cuenta de AWS para obtener más información.
   - Un usuario de la AWS cuenta con permisos para usar Amazon GameLift y otros AWS servicios necesarios. Consulte Configura un Cuenta de AWS para obtener instrucciones sobre cómo configurar un usuario AWS Identity and Access Management (IAM) con GameLift permisos de Amazon.

- Credenciales de su usuario de AWS. Este usuario también necesita un acceso programático con credenciales a largo plazo. Estas credenciales se componen de un ID de clave de acceso de AWS y una clave secreta de AWS. Consulte Obtención de claves de acceso para obtener más información.
- Región AWS. Se trata de la ubicación geográfica en la que desea crear sus recursos de AWS de alojamiento. Durante el desarrollo, te recomendamos que utilices una región cercana a tu ubicación física para minimizar la latencia. Consulte la lista de regiones de AWS admitidas.

4. Cuando hayas seleccionado o creado un perfil, comprueba el estado de arranque del perfil y toma las medidas necesarias. Todos los perfiles deben estar inicializados para poder utilizar la funcionalidad del GameLift plugin de Amazon.

Para arrancar el perfil, tenga en cuenta la siguiente información:

Bootstrapping designa un bucket de Amazon S3 para usarlo con el perfil de usuario seleccionado. Amazon S3 es un AWS servicio fundamental para el almacenamiento de datos y objetos. El depósito que se utiliza para almacenar las configuraciones de los proyectos, crear artefactos y otras dependencias. Los buckets no se comparten entre otros perfiles.

> **Note**
>
> El arranque crea nuevos AWS recursos y puede generar costes.

1. Cuando veas tus perfiles en la ventana del plugin Perfiles AWS de usuario, selecciona el perfil que quieres usar. Aparece un mensaje de advertencia si el perfil aún no se ha iniciado.
2. En la sección Arrancar su perfil, seleccione un perfil de la lista desplegable y compruebe el estado del arranque. Si el estado indica que no existe ningún depósito, pulse el botón Perfil de Bootstrap. Puedes establecer el nombre del depósito en un nombre nuevo, introducir un depósito existente al que tengas acceso o conservar el nombre generado automáticamente.
3. Espere a que el estado del arranque cambie a «Activo». Este proceso puede tardar unos minutos. Cuando el estado es «Activo», puedes usar el perfil para trabajar con las funciones del plugin

Configura tu juego para probarlo localmente con Amazon GameLift Anywhere

En este flujo de trabajo, añades el código de juego de cliente y servidor para las GameLift funciones de Amazon y utilizas el complemento para designar tu estación de trabajo local como host de servidor de juegos de prueba. Cuando haya completado las tareas de integración, utilice el complemento para crear los componentes de cliente y servidor de juegos.

Para iniciar el flujo de trabajo de Amazon GameLift Anywhere:

- En el menú principal del editor de Unity, elige Amazon GameLift y selecciona Host with Anywhere. Esta acción abre la página del plugin para configurar tu juego con una Anywhere flota @. La página presenta un proceso de cinco pasos para integrar, crear y lanzar los componentes del juego.

Configura tu perfil

Elija el perfil que desee utilizar al seguir este flujo de trabajo. El perfil que seleccione afectará a todos los pasos del flujo de trabajo. Todos los recursos que cree estarán asociados a la cuenta de AWS del perfil y se colocarán en la región de AWS predeterminada del perfil. Los permisos del usuario del perfil determinan su acceso a los recursos y acciones de AWS.

1. Seleccione un perfil de la lista desplegable de perfiles disponibles. Si aún no tienes un perfil o quieres crear uno nuevo, ve al GameLift menú de Amazon y selecciona Establecer perfiles de AWS cuenta.
2. Si el estado del arranque no es «Activo», seleccione Perfil de arranque y espere a que el estado cambie a «Activo».

Integra tu juego con Amazon GameLift

> **Note**
>
> Si has importado el juego de muestra, puedes saltarte este paso. Los recursos del juego de muestra ya tienen el código de servidor y cliente necesario.

Para este paso del flujo de trabajo, debes actualizar el código del cliente y del servidor de tu proyecto de juego.

- * Los servidores del juego deben poder comunicarse con el GameLift servicio de Amazon para recibir instrucciones para iniciar una sesión de juego, proporcionar información de conexión de la sesión de juego e informar del estado.
- Los clientes del juego deben poder obtener información sobre las sesiones de juego, unirse a ellas o iniciarlas y obtener información de conexión para unirse a una partida.

Integra tu código de servidor

Si utilizas tu propio proyecto de juego con escenas personalizadas, utiliza el código de ejemplo proporcionado para añadir el código de servidor necesario a tu proyecto de juego:

1. Abre la `Assets/Scripts/Server` carpeta en los archivos del proyecto del juego. Si no existe, créala.
2. Ve al GitHub repositorio aws/ amazon-gamelift-plugin-unity y abre la ruta. `Samples~/SampleGame/Assets/Scripts/Server`
3. Busca el archivo GameLiftServer .cs. y cópialo en la carpeta del servidor de tu proyecto de juego. Cuando crees un ejecutable de servidor, usa este archivo como destino de compilación.

El código de ejemplo incluye estos elementos mínimos obligatorios, que utilizan el SDK del servidor Amazon GameLift C# (versión 5):

- Inicializa un cliente de GameLift API de Amazon. La `InitSDK()` llamada con los parámetros del servidor es obligatoria para una flota de Amazon GameLift Anywhere. Estos ajustes se configuran automáticamente para su uso en el complemento.
- Implementa las funciones de devolución de llamada necesarias para responder a las solicitudes del GameLift servicio de Amazon, incluidas `OnStartGameSession` `OnProcessTerminate`, yon`HealthCheck`.
- Llama `ProcessReady()` con un puerto designado para notificar al GameLift servicio de Amazon cuando el proceso del servidor está listo para albergar sesiones de juego.

Si quieres personalizar el código de servidor de muestra, consulta estos recursos:

- Adición de Amazon GameLift al servidor de juegos
- Referencia del SDK del servidor de Amazon GameLift 5.x para C# y Unity

Integre su código de cliente

Si utilizas tu propio proyecto de juego con escenas personalizadas, necesitas integrar las funciones básicas en el cliente del juego. También debes añadir elementos de interfaz de usuario para que los jugadores puedan iniciar sesión y unirse a una sesión de juego. Usa las API de GameLift servicio de Amazon (en el AWS SDK) para obtener información sobre las sesiones de juego, crear nuevas sesiones de juego o unirte a las sesiones de juego existentes,

Al crear un cliente para pruebas locales con una flota de Anywhere, puedes añadir llamadas directas al GameLift servicio de Amazon. Cuando desarrolle su juego para el alojamiento en la nube, o si planea usar Anywhere Fleets para el alojamiento de producción, necesitará crear un servicio de back-end del lado del cliente para gestionar todas las comunicaciones entre los clientes del juego y el servicio de Amazon. GameLift

Para GameLift integrar Amazon en tu código de cliente, utiliza los siguientes recursos como guía.

- Integre el cliente con la GameLiftCoreApi clase en el GitHub repositorio amazon-gamelift-plugin-unity aws/. Esta clase proporciona controles para la autenticación de los jugadores y para recuperar la información de la sesión del juego.
- Consulta ejemplos de integraciones de juegos, disponibles en el GitHub repositorio aws/,. amazon-gamelift-plugin-unity `Samples~/SampleGame/Assets/Scripts/Client/GameLiftClient.cs`
- Sigue las instrucciones de Añadir Amazon GameLift a tu cliente de juegos de Unity.

Para los clientes de juegos que se conecten a una flota de Anywhere, tu cliente de juego necesita la siguiente información. El complemento actualiza automáticamente tu proyecto de juego para usar los recursos que crees en el complemento.

- FleetId - El identificador único de tu flota de Anywhere.
- FleetLocation - La ubicación personalizada de su flota de Anywhere.
- AwsRegion - La AWS región en la que está alojada tu flota de Anywhere. Esta es la región que configuraste en tu perfil de usuario.
- ProfileName - Un perfil de AWS credenciales en su máquina local que le permita acceder al AWS SDK GameLift. El cliente del juego usa estas credenciales para autenticar las solicitudes al GameLift servicio de Amazon.

> **Note**
>
> El perfil de credenciales lo genera el complemento y lo almacena en la máquina local. Como resultado, debe ejecutar el cliente en la máquina local (o en una máquina con el mismo perfil).

Conéctese a una flota en cualquier lugar

En este paso, tendrá que designar la flota de Anywhere que desee utilizar. Una flota de Anywhere define un conjunto de recursos informáticos, que se pueden ubicar en cualquier lugar para el alojamiento de servidores de juegos.

- Si la AWS cuenta que utilizas actualmente tiene flotas Anywhere existentes, abre el campo desplegable del nombre de la flota y elige una flota. Este menú desplegable solo muestra las flotas de Anywhere de la región de AWS para el perfil de usuario activo actualmente.
- Si no hay ninguna flota existente, o si deseas crear una nueva, selecciona Crear nueva flota en cualquier lugar y proporciona un nombre de flota.

Una vez que hayas elegido una flota de Anywhere para tu proyecto, Amazon GameLift verifica que el estado de la flota esté activo y muestra el identificador de la flota. Puedes hacer un seguimiento del progreso de esta solicitud en el registro de resultados del editor de Unity.

Registra un cómputo

En este paso, se registrará su estación de trabajo local como recurso informático en la nueva flota de Anywhere.

1. Especifique un nombre de equipo para la máquina local. Si añade más de un recurso informático a la flota, los nombres deben ser únicos.
2. Seleccione Registrar cómputo. Puede realizar un seguimiento del progreso de esta solicitud en el registro de resultados del editor de Unreal.

El complemento registra su estación de trabajo local con la dirección IP establecida en localhost (127.0.0.1). Esta configuración supone que ejecutarás el cliente y el servidor del juego en la misma máquina.

En respuesta a esta acción, Amazon GameLift comprueba que puede conectarse al equipo y devuelve información sobre el equipo recién registrado.

Inicia el juego

En este paso, construyes los componentes del juego y los lanzas para jugar. Realice los siguientes pasos:

1. Configura tu cliente de juego. En este paso, le pides al complemento que actualice un `GameLiftClientSettings` activo para tu proyecto de juego. El complemento utiliza este recurso para almacenar cierta información que el cliente del juego necesita para conectarse al GameLift servicio de Amazon.

    a. Si no has importado ni inicializado el juego de muestra, crea un nuevo `GameLiftClientSettings` activo. En el menú principal del editor de Unity, selecciona Activos GameLift, Crear y Configuración del cliente. Si creas varias copias GameLiftClientSettings en tu proyecto, el plugin lo detectará automáticamente y te notificará qué activo actualizará el plugin.

    b. En Launch Game, selecciona Configure Client: Apply Anywhere Settings. Esta acción actualiza la configuración del cliente del juego para usar la flota Anywhere que acabas de configurar.

2. Crea y ejecuta tu cliente de juego.

    a. Crea un cliente ejecutable mediante el proceso de compilación estándar de Unity. En Archivo, Configuración de compilación, cambia la plataforma a Windows, Mac o Linux. Si has importado el juego de muestra e inicializado la configuración, la lista de compilaciones y el objetivo de la compilación se actualizan automáticamente.

    b. Lanza una o más instancias del ejecutable del cliente de juego recién creado.

3. Lanza un servidor de juegos en tu flota de Anywhere. Elige un servidor: inicia el servidor en el editor. Esta tarea inicia un servidor activo al que su cliente puede conectarse mientras el editor de Unity permanezca abierto.

4. Inicia o únete a una sesión de juego. En tus instancias de cliente de juego, usa la interfaz de usuario para unir a cada cliente a una sesión de juego. La forma de hacerlo depende de la forma en que se hayan añadido funciones al cliente.

Si utilizas el cliente de juego de muestra, tiene las siguientes características:

- Un componente de inicio de sesión para jugadores. Al conectarte a un servidor de juegos de una flota de Anywhere, no se valida el jugador. Puedes introducir cualquier valor para unirte a la sesión de juego.
- Una sencilla interfaz de usuario para unirse al juego. Cuando un cliente intenta unirse a una partida, busca automáticamente una sesión de juego activa con un espacio de jugador disponible. Si no hay ninguna sesión de juego disponible, el cliente solicita una nueva sesión de juego. Si hay una sesión de juego disponible, el cliente solicita unirse a la sesión de juego disponible. Al probar el juego con varios clientes simultáneos, el primer cliente inicia la sesión de juego y los demás se unen automáticamente a la sesión de juego existente.
- Sesiones de juego con ranuras para cuatro jugadores. Puedes lanzar hasta cuatro instancias de clientes de juego simultáneamente y se unirán a la misma sesión de juego.

Lánzalo desde un ejecutable de servidor (opcional)

Puedes crear e iniciar el ejecutable de tu servidor de juegos para probarlo en una flota de Anywhere.

1. Cree un servidor ejecutable mediante el proceso de compilación estándar de Unity. En Archivo, Configuración de compilación, cambie la plataforma a Servidor dedicado y compile.
2. Obtenga un token de autenticación a corto plazo llamando al comando AWS CLI get-compute-auth-tokencon su ID de flota y AWS región de Anywhere. El identificador de la flota se muestra en Connect to an Anywhere Fleet al crear la flota. La AWS región se muestra en Establezca su perfil al seleccionar su perfil activo.

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

3. Ejecuta el servidor de juegos recién creado desde una línea de comandos e introduce un token de autenticación válido.

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

Implementación del juego en un alojamiento en la nube con flotas de EC2 administradas

En este flujo de trabajo, utilizas el complemento para preparar el juego para su alojamiento en recursos informáticos basados en la nube gestionados por Amazon GameLift. Añades el código del

juego de cliente y servidor para la GameLift funcionalidad de Amazon y, a continuación, subes la versión de tu servidor al GameLift servicio de Amazon para su alojamiento. Cuando se complete este flujo de trabajo, dispondrás de servidores de juegos en la nube y de un cliente de juego funcional que podrá conectarse a ellos.

Para iniciar el flujo de trabajo de Amazon EC2 GameLift gestionado por Amazon:

- En el menú principal del editor de Unity, elige Amazon GameLift y selecciona Host with Managed EC2. Este flujo de trabajo presenta un proceso de seis pasos para integrar, crear, implementar y lanzar los componentes del juego.

Configura tu perfil

Elija el perfil que desee utilizar al seguir este flujo de trabajo. El perfil que seleccione afectará a todos los pasos del flujo de trabajo. Todos los recursos que cree estarán asociados a la cuenta de AWS del perfil y se colocarán en la región de AWS predeterminada del perfil. Los permisos del usuario del perfil determinan su acceso a los recursos y acciones de AWS.

1. Seleccione un perfil de la lista desplegable de perfiles disponibles. Si aún no tienes un perfil o quieres crear uno nuevo, ve al GameLift menú de Amazon y selecciona Establecer perfiles de AWS cuenta.
2. Si el estado del arranque no es «Activo», seleccione Perfil de arranque y espere a que el estado cambie a «Activo».

Integra tu juego con Amazon GameLift

Para esta tarea, debes actualizar el código del cliente y del servidor en tu proyecto de juego.

- Los servidores de juegos deben poder comunicarse con el GameLift servicio de Amazon para recibir instrucciones para iniciar una sesión de juego, proporcionar información de conexión de la sesión de juego e informar del estado.
- Los clientes del juego deben poder obtener información sobre las sesiones de juego, unirse a ellas o iniciarlas y obtener información de conexión para unirse a una partida.

> **Note**
>
> Si has importado el juego de muestra, puedes saltarte este paso. Los recursos del juego de muestra ya tienen el código de servidor y cliente necesario.

Integra tu código de servidor

Cuando utilices tu propio proyecto de juego con escenas personalizadas, utiliza el código de ejemplo proporcionado para añadir el código de servidor necesario a tu proyecto de juego. Si integraste tu proyecto de juego para probarlo con una flota de Anywhere, ya has completado las instrucciones de este paso.

1. Abre la `Assets/Scripts/Server` carpeta en los archivos del proyecto del juego. Si no existe, créala.
2. Ve al GitHub repositorio aws/ amazon-gamelift-plugin-unity y abre la ruta. `Samples~/SampleGame/Assets/Scripts/Server`
3. Localiza el archivo `GameLiftServer.cs` y cópialo en la carpeta de tu proyecto de juego. `Server` Cuando crees un ejecutable de servidor, usa este archivo como destino de compilación.

El código de ejemplo incluye estos elementos mínimos obligatorios, que utilizan el SDK del servidor Amazon GameLift C# (versión 5):

- Inicializa un cliente de GameLift API de Amazon. La llamada initSDK () con los parámetros del servidor es necesaria para una flota de Amazon GameLift Anywhere. Estos ajustes se configuran automáticamente para su uso en el complemento.
- Implementa las funciones de devolución de llamada necesarias para responder a las solicitudes del GameLift servicio de Amazon, incluidas `OnStartGameSessionOnProcessTerminate`, yon`HealthCheck`.
- Llama `ProcessReady()` con un puerto designado para notificar al GameLift servicio de Amazon cuando el proceso del servidor está listo para albergar sesiones de juego.

Si quieres personalizar el código de servidor de muestra, consulta estos recursos:

- Adición de Amazon GameLift al servidor de juegos
- Referencia del SDK del servidor de Amazon GameLift 5.x para C# y Unity

Integre su código de cliente

Para los clientes de juegos que se conectan a servidores de juegos basados en la nube, se recomienda utilizar un servicio de back-end del lado del cliente para realizar llamadas al GameLift servicio de Amazon, en lugar de hacerlo directamente desde el cliente del juego.

En el flujo de trabajo de los complementos para el alojamiento en una flota de EC2 gestionada, cada escenario de despliegue incluye un servicio de backend prediseñado que incluye los siguientes componentes:

- Conjunto de funciones Lambda y tablas de DynamoDB que se utilizan para solicitar sesiones de juego y recuperar información sobre las sesiones de juego. Estos componentes utilizan una puerta de enlace API como proxy.
- Un grupo de usuarios de Amazon Cognito que genera identificadores de jugador únicos y autentica las conexiones de los jugadores.

Para utilizar estos componentes, el cliente del juego necesita una funcionalidad que le permita enviar solicitudes al servicio de back-end para hacer lo siguiente:

- Crea un usuario jugador en el grupo de usuarios de AWS Cognito y autentícate.
- Únase a una sesión de juego y reciba información de conexión.
- Únete a una partida mediante el matchmaking.

Usa los siguientes recursos como guía.

- Integre el cliente con la GameLiftCoreApiclase en el GitHub repositorio amazon-gamelift-plugin-unityaws/. Esta clase proporciona controles para la autenticación de los jugadores y para recuperar la información de la sesión del juego.
- Para ver los ejemplos de integraciones de juegos, ve al GitHub repositorio aws/,. amazon-gamelift-plugin-unity `Samples~/SampleGame/Assets/Scripts/Client/GameLiftClient.cs`
- Agrega Amazon GameLift a tu cliente de juegos de Unity.

Selecciona el escenario de despliegue

En este paso, tendrá que elegir la solución de alojamiento de juegos que desee implementar en ese momento. Puede disponer de varias implementaciones del juego mediante cualquiera de los escenarios.

- Flota de una sola región: despliega tu servidor de juegos en una sola flota de recursos de alojamiento en la región predeterminada AWS del perfil activo. Este escenario es un buen punto de partida para probar la integración del servidor con AWS y la configuración de compilación del servidor. Permite implementar los siguientes recursos:
  - La flota de AWS (bajo demanda) con la compilación del servidor de juegos instalada y en ejecución.
  - Grupo de usuarios y cliente de Amazon Cognito para permitir a los jugadores autenticarse e iniciar un juego.
  - Autorizador de la puerta de enlace de API que vincula el grupo de usuarios con las API.
  - WebACI para limitar las llamadas excesivas de los jugadores a la puerta de enlace de la API.
  - Puerta de enlace de la API + función de Lambda para que los jugadores soliciten una ranura de juego. Esta función llama a `CreateGameSession()` si no hay ninguna disponible.
  - Puerta de enlace de la API + función de Lambda para que los jugadores obtengan información de la conexión para su solicitud de juego.
- FlexMatch flota: despliega tu servidor de juego en un conjunto de flotas y configura un FlexMatch emparejador con reglas para crear partidas de jugadores. En este escenario, se utiliza un alojamiento Spot de bajo coste con una estructura de varias flotas y ubicaciones para garantizar una disponibilidad duradera. Este enfoque resulta útil cuando estás listo para empezar a diseñar un componente de emparejamiento para tu solución de alojamiento. En este escenario, crearás los recursos básicos para esta solución, que podrás personalizar más adelante según sea necesario. Permite implementar los siguientes recursos:
  - FlexMatch Configuración y reglas de emparejamiento establecidas para aceptar las solicitudes de los jugadores y formar partidas.
  - Tres flotas de AWS con la compilación del servidor de juegos instalada y en ejecución en varios lugares. Incluye dos flotas de spot y una flota bajo demanda como respaldo.
  - Cola de ubicación de sesión de juego de AWS que responde a las solicitudes de emparejamientos propuestos mediante la búsqueda del mejor recurso de alojamiento posible (en función de la viabilidad, el costo, la latencia de los jugadores, etc.) y el inicio de una sesión de juego.
  - Grupo de usuarios y cliente de Amazon Cognito para permitir a los jugadores autenticarse e iniciar un juego.
  - Autorizador de la puerta de enlace de API que vincula el grupo de usuarios con las API.
  - WebACI para limitar las llamadas excesivas de los jugadores a la puerta de enlace de la API.

- Puerta de enlace de la API + función de Lambda para que los jugadores soliciten una ranura de juego. Esta función llama a `StartMatchmaking()`.
- Puerta de enlace de la API + función de Lambda para que los jugadores obtengan información de la conexión para su solicitud de juego.
- Tablas de Amazon DynamoDB para almacenar tickets de emparejamiento para jugadores e información sobre las sesiones de juego.
- Tema de SNS más función Lambda para GameSessionQueue gestionar eventos.

Establece los parámetros del juego

En este paso, deberá describir el juego que quiera subir a AWS.

- Nombre del juego: proporciona un nombre significativo para tu proyecto de juego. Este nombre se usa en el complemento.
- Nombre de la flota: proporcione un nombre significativo para la flota de EC2 gestionada. Amazon GameLift usa este nombre (junto con el identificador de flota) al publicar recursos en la AWS consola.
- Nombre de compilación: proporciona un nombre significativo para la compilación de tu servidor. AWSusa este nombre para hacer referencia a la copia de la versión de tu servidor que se carga en Amazon GameLift y se usa para las implementaciones.
- Parámetros de lanzamiento: introduzca las instrucciones opcionales que se ejecutarán al lanzar el ejecutable del servidor en una instancia de flota de EC2 gestionada. La longitud máxima es de 1024 caracteres.
- Carpeta del servidor del juego: proporciona la ruta a la carpeta local que contiene la versión del servidor.
- Archivo del servidor del juego: especifique el nombre del archivo ejecutable del servidor.

Escenario de despliegue

En este paso deberá implementar el juego en una solución de alojamiento en la nube en función del escenario de implementación que elija. Este proceso puede tardar hasta 40 minutos y, además, AWS valida la compilación del servidor, realiza un aprovisionamiento de los recursos de alojamiento, instala el servidor de juegos y lanza los procesos del servidor y los prepara para alojar sesiones de juego.

Para iniciar la implementación, elija Implementar CloudFormation. Puede realizar el seguimiento del estado del alojamiento de su juego aquí. Para obtener información más detallada, puede iniciar sesión en la consola de administración de AWS para AWS y ver las notificaciones de eventos. Asegúrese de iniciar sesión con la misma cuenta, usuario y región de AWS que el perfil de usuario activo del complemento.

Cuando se complete la implementación, tendrá el servidor de juegos instalado en una instancia de EC2 de AWS. Hay al menos un proceso del servidor en ejecución y listo para iniciar una sesión de juego.

Inicie el cliente del juego

Cuando tu flota se haya desplegado correctamente, dispondrás de servidores de juegos en funcionamiento y disponibles para albergar sesiones de juego. Ahora puedes crear tu cliente, lanzarlo y conectarte para unirte a la sesión de juego.

1. Configura tu cliente de juego. En este paso, le pides al complemento que actualice un `GameLiftClientSettings` activo para tu proyecto de juego. El complemento utiliza este recurso para almacenar cierta información que el cliente del juego necesita para conectarse al GameLift servicio de Amazon.

    a. Si no has importado ni inicializado el juego de muestra, crea un nuevo `GameLiftClientSettings` activo. En el menú principal del editor de Unity, selecciona Activos GameLift, Crear y Configuración del cliente. Si creas varias copias GameLiftClientSettings en tu proyecto, el plugin lo detectará automáticamente y te notificará qué activo actualizará el plugin.

    b. En Launch Game, seleccione Configurar cliente: aplicar la configuración de EC2 gestionada. Esta acción actualiza la configuración del cliente del juego para que utilice la flota de EC2 gestionada que acaba de implementar.

2. Crea tu cliente de juego. Cree un cliente ejecutable mediante el proceso de compilación estándar de Unity. En Archivo, Configuración de compilación, cambia la plataforma a Windows, Mac o Linux. Si has importado el juego de muestra e inicializado la configuración, la lista de compilaciones y el objetivo de la compilación se actualizan automáticamente.
3. Abre el ejecutable del cliente del juego recién creado. Para empezar a jugar, inicia de dos a cuatro instancias de cliente y usa la interfaz de usuario de cada una para unirte a una sesión de juego.

Si utilizas el cliente de juego de muestra, tiene las siguientes características:

- Un componente de inicio de sesión para jugadores. Al conectarte a un servidor de juegos de una flota de Anywhere, no se valida el jugador. Puedes introducir cualquier valor para unirte a la sesión de juego.
- Una sencilla interfaz de usuario para unirse al juego. Cuando un cliente intenta unirse a una partida, busca automáticamente una sesión de juego activa con un espacio de jugador disponible. Si no hay ninguna sesión de juego disponible, el cliente solicita una nueva sesión de juego. Si hay una sesión de juego disponible, el cliente solicita unirse a la sesión de juego disponible. Al probar el juego con varios clientes simultáneos, el primer cliente inicia la sesión de juego y los demás se unen automáticamente a la sesión de juego existente.
- Sesiones de juego con ranuras para cuatro jugadores. Puedes lanzar hasta cuatro instancias de clientes de juego simultáneamente y se unirán a la misma sesión de juego.

Guía GameLift del complemento de Amazon para Unity para el SDK 4.x del servidor

> **Note**
>
> En este tema se proporciona información sobre una versión anterior del GameLift complemento Amazon para Unity. La versión 1.0.0 (publicada en 2021) usa el SDK 4.x o anterior para GameLift servidores de Amazon. Para obtener documentación sobre la última versión del complemento, que utiliza el SDK 5.x del servidor y es compatible con Amazon GameLift Anywhere, consulte Guía GameLift del complemento de Amazon para Unity para el SDK de servidor 5.x.

Amazon GameLift proporciona herramientas para preparar tus servidores de juegos multijugador para que se ejecuten en Amazon GameLift. El GameLift complemento Amazon para Unity facilita la integración de Amazon GameLift en tus proyectos de juegos de Unity y el despliegue de GameLift los recursos de Amazon para el alojamiento en la nube. Usa el complemento de Unity para acceder a GameLift las API de Amazon e implementar AWS CloudFormation plantillas para escenarios de juego comunes.

Una vez que hayas configurado el complemento, puedes probar el ejemplo de Amazon GameLift Unity en GitHub.

Temas

- Integre Amazon GameLift con un proyecto de servidor de juegos de Unity
- Integre Amazon GameLift con un proyecto de cliente de juegos de Unity
- Instala y configura el complemento
- Prueba tu juego localmente
- Despliega un escenario
- Integra juegos con Amazon GameLift en Unity
- Importa y ejecuta un juego de muestra

Integre Amazon GameLift con un proyecto de servidor de juegos de Unity

> **Note**
>
> En este tema se proporciona información sobre una versión anterior del GameLift complemento Amazon para Unity. La versión 1.0.0 (publicada en 2021) usa el SDK 4.x o anterior para GameLift servidores de Amazon. Para obtener documentación sobre la última versión del complemento, que utiliza el SDK 5.x del servidor y es compatible con Amazon GameLift Anywhere, consulteGuía GameLift del complemento de Amazon para Unity para el SDK de servidor 5.x.

Este tema te ayuda a preparar tu servidor de juegos personalizado para alojarlo en Amazon GameLift. El servidor del juego debe poder notificar a Amazon GameLift su estado, iniciar y detener las sesiones de juego cuando se le solicite y realizar otras tareas. Para obtener más información, consulte Adición de Amazon GameLift al servidor de juegos.

Requisitos previos

Antes de integrar el servidor de juegos, realice las siguientes tareas:

- Configurar un rol de servicio de IAM para Amazon GameLift
- Instala el complemento para Unity

Configuración de un proceso del servidor nuevo

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

Establece la comunicación con Amazon GameLift e informa que el proceso del servidor está listo para albergar una sesión de juego.

1. Inicie el SDK del servidor mediante una llamada a `InitSDK()`
2. Para preparar el servidor para que acepte una sesión de juego, llame a `ProcessReady()` con el puerto de conexión y los detalles de la ubicación de la sesión de juego. Incluye los nombres de las funciones de devolución de llamada que invoca el GameLift servicio de Amazon, como`OnGameSession()`,`OnGameSessionUpdate()`,`OnProcessTerminate()`. `OnHealthCheck()` Amazon GameLift puede tardar unos minutos en devolverte la llamada.
3. Amazon GameLift actualiza el estado del proceso del servidor a`ACTIVE`.
4. Amazon GameLift llama on`HealthCheck` periódicamente.

El siguiente ejemplo de código muestra cómo configurar un proceso de servidor sencillo con 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;
}
```

Inicio de una sesión de juego

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

Una vez completada la inicialización del juego, podrá iniciar una sesión de juego.

1. Implemente la función de devolución de llamada `onStartGameSession`. Amazon GameLift invoca este método para iniciar una nueva sesión de juego en el proceso del servidor y recibir las conexiones de los jugadores.

2. Para activar una sesión de juego, llame a `ActivateGameSession()`. Para obtener más información sobre los SDK, consulte Referencia del SDK del servidor de Amazon GameLift (C#): Acciones.

El siguiente ejemplo de código ilustra cómo iniciar una sesión de juego con 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();
}
```

Finalización de una sesión de juego

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

Notifica a Amazon GameLift cuando finalice una sesión de juego. Como práctica recomendada, cierre los procesos del servidor una vez finalizada la sesión de juego para reciclar y actualizar los recursos de alojamiento.

1. Configura una función llamada on`ProcessTerminate` para recibir solicitudes de Amazon GameLift y llama`ProcessEnding()`.
2. El estado del proceso cambia a `TERMINATED`.

El siguiente ejemplo describe cómo finalizar un proceso de una sesión de juego.

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

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

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

Crear un servidor, compilarlo y subirlo a Amazon GameLift

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

Tras integrar tu servidor de juegos con Amazon GameLift, sube los archivos de compilación a una flota para que Amazon GameLift pueda desplegarlos como alojamiento de juegos. Para obtener más información sobre cómo subir tu servidor a Amazon GameLift, consultaCarga de una compilación del servidor de juegos personalizada en Amazon GameLift.

Integre Amazon GameLift con un proyecto de cliente de juegos de Unity

> **Note**
>
> En este tema se proporciona información sobre una versión anterior del GameLift complemento Amazon para Unity. La versión 1.0.0 (publicada en 2021) usa el SDK 4.x o anterior para GameLift servidores de Amazon. Para obtener documentación sobre la última versión del complemento, que utiliza el SDK 5.x del servidor y es compatible con Amazon GameLift Anywhere, consulteGuía GameLift del complemento de Amazon para Unity para el SDK de servidor 5.x.

Este tema te ayuda a configurar un cliente de juego para conectarse a las sesiones de juego GameLift alojadas en Amazon a través de un servicio de back-end. Usa GameLift las API de Amazon para iniciar el matchmaking, solicitar la ubicación de las sesiones de juego y mucho más.

Añade código al proyecto de servicio de backend para permitir la comunicación con el GameLift servicio de Amazon. Un servicio de back-end gestiona todas las comunicaciones del cliente del juego con el GameLift servicio. Para obtener más información sobre los servicios de backend, consulte Diseño del servicio de cliente de juegos.

El servicio de backend se encarga de las siguientes tareas de cliente de juegos:

- Personalización de la autenticación de sus jugadores..

- Solicita información sobre las sesiones de juego activas al GameLift servicio de Amazon.
- Creación de una sesión de juego nueva.
- Adición de un jugador a una sesión de juego existente.
- Eliminación de un jugador de una sesión de juego existente.

Temas

Requisitos previos

Antes de configurar la comunicación del servidor del juego con el GameLift cliente de Amazon, realiza las siguientes tareas:

- Configura un Cuenta de AWS
- Instala el complemento para Unity
- Integre Amazon GameLift con un proyecto de servidor de juegos de Unity
- Configuración de las flotas de Amazon GameLift

Inicialización de un cliente de juegos

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

Añada código para inicializar un cliente de juegos. Ejecuta este código al lanzarlo, es necesario para otras GameLift funciones de Amazon.

1. Inicialice `AmazonGameLiftClient`. Llame a `AmazonGameLiftClient` con una configuración de cliente predeterminada o con una configuración personalizada. Para obtener más información

sobre cómo configurar un cliente, consulte Configurar Amazon GameLift en un servicio de back-end.

2. Genere un ID de jugador único para que cada jugador se conecte a una sesión de juego. Para más información, consulte Generación de ID de jugador.

   Los siguientes ejemplos muestran cómo configurar un GameLift cliente de 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);

    }
}
```

Creación de una sesión de juego en una flota específica

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

Añada código para iniciar sesiones de juego nuevas en las flotas implementadas y ponerlas a disposición de los jugadores. Cuando Amazon GameLift haya creado la nueva sesión de juego y haya devuelto una`GameSession`, podrás añadirle jugadores.

- Realice una solicitud para una nueva sesión de juego.
  - Si el juego utiliza flotas, llame a `CreateGameSession()` con un ID de la flota o alias, un nombre de sesión y el número máximo de jugadores simultáneos para el juego.
  - Si el juego utiliza colas, llame a `StartGameSessionPlacement()`

En el siguiente ejemplo se muestra cómo crear una sesión de juego.

```
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;
}
```

Adición de jugadores a las sesiones de juego

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

Cuando Amazon GameLift haya creado la nueva sesión de juego y haya devuelto un `GameSession` objeto, podrás añadirle jugadores.

1. Reserve una ranura de jugador en una sesión de juego mediante la creación de una sesión de jugador nueva. Utilice `CreatePlayerSession` o `CreatePlayerSessions` con el ID de sesión de juego y un ID único para cada jugador.
2. Conéctese a la sesión de juego. Recupere el objeto `PlayerSession` para obtener la información de conexión de la sesión de juego. Puede utilizar esta información para establecer una conexión directa con el proceso del servidor:
   a. Utilice el puerto especificado y el nombre de DNS o la dirección IP del proceso del servidor.
   b. Utilice el nombre y el puerto DNS de sus flotas. El nombre y el puerto DNS son obligatorios si las flotas tienen la generación del certificado TLS habilitada.
   c. Haga referencia al ID de sesión del jugador. Si el servidor de juegos valida las conexiones de los jugadores entrantes, el ID de sesión del jugador será obligatorio.

Los siguientes ejemplos muestran cómo reservar un spot de jugador en una sesión de juego.

```
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;
```

```
}
```

El siguiente código muestra cómo conectar a un jugador con la sesión de juego.

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

Eliminación de un jugador de una sesión de juego

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

Puedes eliminar a los jugadores de la sesión de juego cuando abandonen el juego.

1. Notifica al GameLift servicio de Amazon que un jugador se ha desconectado del proceso del servidor. Llame a `RemovePlayerSession` con el ID de sesión del jugador.
2. Verifique que `RemovePlayerSession` devuelva `Success`. Luego, Amazon GameLift cambia el espacio del jugador para que esté disponible, que Amazon GameLift puede asignar a un nuevo jugador.

En el ejemplo siguiente se muestra cómo eliminar una sesión de jugador.

```
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());
    }
}
```

Instala y configura el complemento

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

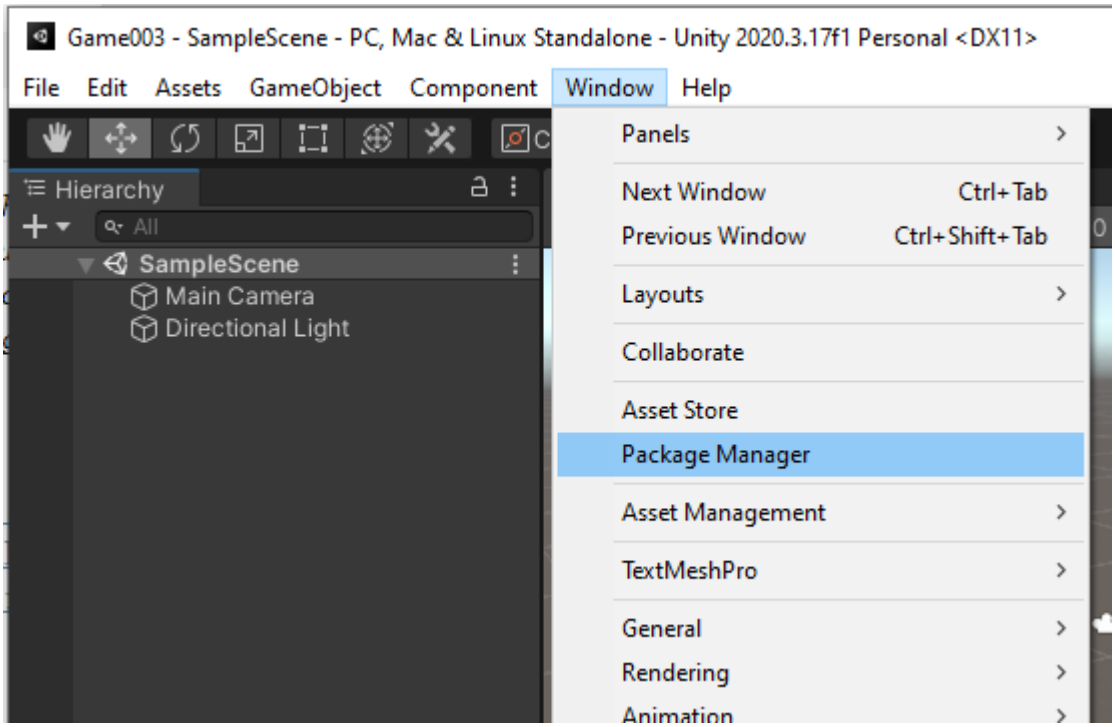
En esta sección se describe cómo descargar, instalar y configurar el GameLift complemento Amazon para Unity, versión 1.0.0.

Requisitos previos

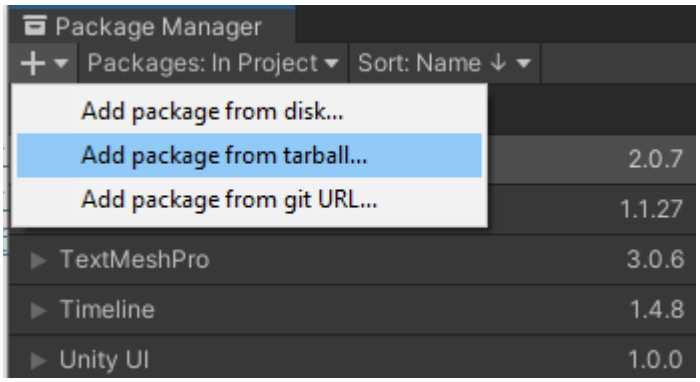
- Unity para Windows 2019.4 LTS, Windows 2020.3 LTS o Unity para MacOS
- Versión actual de Java
- Versión actual de .NET 4.x

Para descargar e instalar el complemento para Unity, realice el siguiente procedimiento:

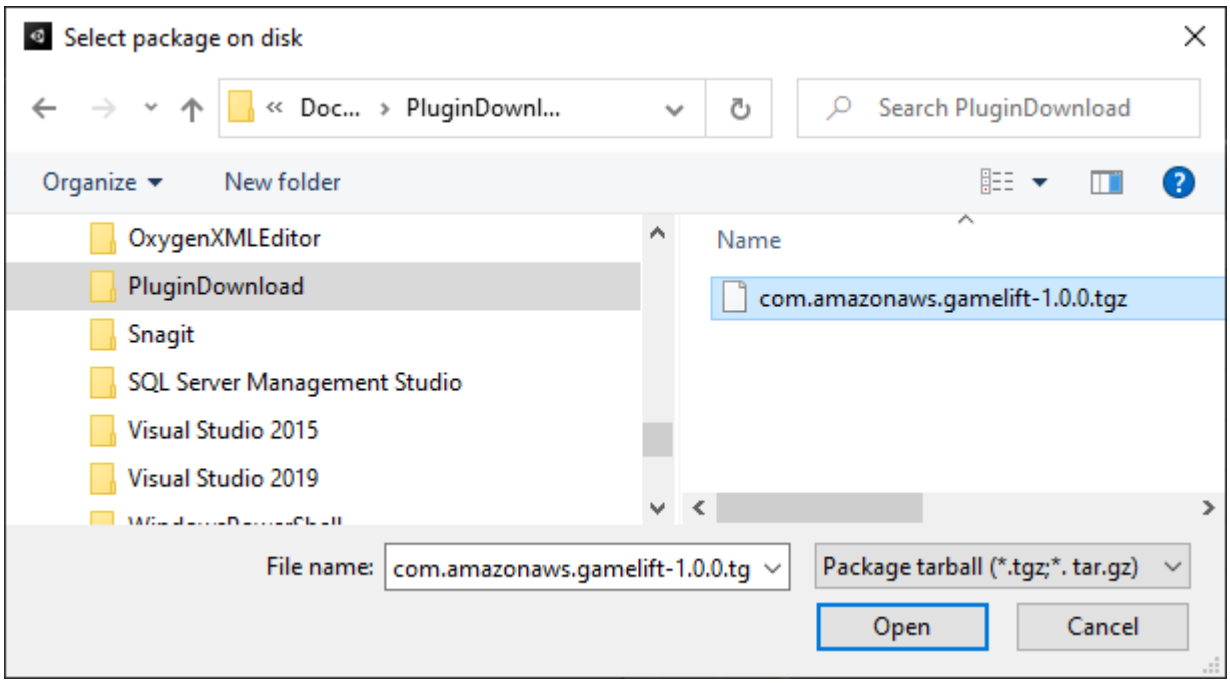
1. Descarga el GameLift plugin de Amazon para Unity. Puedes encontrar la última versión en la página del repositorio del GameLift plugin de Amazon para Unity. En la versión más reciente, elija Activos y, a continuación, descargue el archivo `com.amazonaws.gamelift-`*`version`*`.tgz`.
2. Lance Unity y elija un proyecto.
3. En la barra de navegación superior, en Ventana, elija Administrador de paquetes:



4. En la pestaña Administrador de paquetes, seleccione + y, a continuación, elija Añadir paquete desde archivo tarball... :



5. En la ventana Seleccionar paquetes en disco, navegue hasta la `com.amazonaws.gamelift` carpeta, elija el archivo y`com.amazonaws.gamelift-version.tgz` , a continuación, elija Abrir:



6. Una vez que Unity haya cargado el complemento, Amazon GameLift aparecerá como un elemento nuevo en el menú de Unity. La instalación y recopilación de los scripts puede llevar unos minutos. La pestaña Amazon GameLift Plugin Settings se abre automáticamente.

7. En el panel del SDK, elija Usar .NET 4.x.

   Cuando esté configurado, el estado cambiará de No configurado a Configurado.

Prueba tu juego localmente

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

Usa Amazon GameLift Local para ejecutar Amazon GameLift en tu dispositivo local. Puedes usar Amazon GameLift Local para verificar los cambios de código en cuestión de segundos, sin conexión de red.

Configure las pruebas locales

1. En la ventana del complemento para Unity, elija la pestaña Prueba.
2. En el panel de pruebas, selecciona Descargar Amazon GameLift Local. El complemento para Unity abrirá una ventana del navegador y descargará el archivo `GameLift_06_03_2021.zip` en la carpeta de descargas.

   La descarga incluye el SDK del servidor de C#, los archivos de origen de.NET y un componente de .NET compatible con Unity.
3. Descomprima el archivo `GameLift_06_03_2021.zip` descargado.
4. En la ventana de configuración del GameLift plugin de Amazon, selecciona Amazon GameLift Local Path, navega hasta la carpeta descomprimida, selecciona el archivo y**`GameLiftLocal.jar`**, a continuación, selecciona Abrir.

   Cuando esté configurado, el estado de la prueba local cambiará de No configurado a Configurado.
5. Verifique el estado de JRE. Si el estado es No configurado, seleccione Descargar JRE e instale la versión de Java recomendada.

   Después de instalar y configurar el entorno de Java, el estado cambiará a Configurado.

Ejecuta tu juego local

1. En la pestaña del complemento para Unity, elija la pestaña Prueba.
2. En el panel Prueba, elija Abrir interfaz de usuario de prueba local.

3. En la ventana Prueba local, especifique una ruta de un archivo ejecutable del servidor. Elija ... para seleccionar la ruta y el nombre del archivo ejecutable de la aplicación del servidor.
4. En la ventana Prueba local, especifique un puerto local de GL.
5. Elija Implementar y ejecutar para implementar y ejecutar el servidor.
6. Para detener el servidor de juegos, elija Detener o cierre las ventanas del servidor de juegos.

Despliega un escenario

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

Un escenario usa una AWS CloudFormation plantilla para crear los recursos que necesitas para implementar una solución de alojamiento en la nube para tu juego. En esta sección se describen los escenarios que GameLift ofrece Amazon y cómo utilizarlos.

Requisitos previos

Para implementar el escenario, necesitas un rol de IAM para el GameLift servicio de Amazon. Para obtener información sobre cómo crear un rol para Amazon GameLift, consulteConfigura un Cuenta de AWS.

Cada situación requiere permisos para los siguientes recursos:

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

Escenarios

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

El GameLift complemento Amazon para Unity incluye los siguientes escenarios:

Solo autenticación

En este escenario se crea un servicio de backend del juego que realiza la autenticación del jugador sin la capacidad del servidor de juegos. La plantilla crea los siguientes recursos en su cuenta:

- Un grupo de usuarios de Amazon Cognito para almacenar la información de autenticación de los jugadores.
- Un controlador AWS Lambda respaldado por un punto de conexión REST de Amazon API Gateway que inicia los juegos y visualiza la información de conexión del juego.

Flota de una sola región

Este escenario crea un servicio de back-end de juegos con una sola GameLift flota de Amazon. Crea los siguientes recursos:

- Grupo de usuarios de Amazon Cognito para un jugador para permitir a los jugadores autenticarse e iniciar un juego.
- Un controlador AWS Lambda para buscar una sesión de juego existente con una ranura de jugador abierta en la flota. Si no encuentra una ranura abierta, cree una nueva sesión de juego.

Flota de varias regiones con una cola y un emparejador personalizado

Este escenario forma partidas utilizando las GameLift colas de Amazon y un emparejador personalizado para agrupar a los jugadores más antiguos de la piscina de espera. Crea los siguientes recursos:

- Un tema de Amazon Simple Notification Service en el que Amazon GameLift publica mensajes. Para obtener más información sobre los temas y notificaciones de SNS, consulte Configuración de la notificación de eventos para la ubicación de sesiones de juego..

- Una función de Lambda que se invoca mediante el mensaje que comunica los detalles de ubicación y conexión al juego.
- Una tabla de Amazon DynamoDB para almacenar los detalles de ubicación y conexión al juego. Las llamadas a `GetGameConnection` se leen de esta tabla y se devuelve la información de conexión al cliente de juego.

Flotas de spot con una cola y un emparejador personalizado

Este escenario forma coincidencias mediante GameLift colas de Amazon y un emparejador personalizado y configura tres flotas. Crea los siguientes recursos:

- Dos flotas de spot que contienen diferentes tipos de instancias para garantizar la durabilidad en caso de que spot no esté disponible.
- Una flota bajo demanda que actúa como respaldo para las demás flotas de spot. Para obtener más información sobre el diseño de las flotas, consulte Guía de diseño de flotas de Amazon GameLift.
- Una GameLift cola de Amazon para mantener una alta disponibilidad de los servidores y un coste bajo. Para obtener más información sobre las prácticas recomendadas sobre colas, consulte Diseño de colas de sesiones de juego.

FlexMatch

Este escenario utiliza FlexMatch, un servicio de emparejamiento administrado, para unir a los jugadores del juego. Para obtener más información FlexMatch, consulta Qué es Amazon GameLift FlexMatch. En este escenario se crean los siguientes recursos:

- Una función de Lambda para crear un ticket de emparejamiento después de recibir las solicitudes de `StartGame`.
- Una función Lambda independiente para escuchar los eventos de los FlexMatch partidos.

Para evitar cargos innecesarios en su Cuenta de AWS, elimine los recursos creados por cada escenario una vez que haya terminado de usarlos. Elimine la pila de AWS CloudFormation correspondiente.

Actualice AWS las credenciales

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

El GameLift complemento de Amazon para Unity requiere credenciales de seguridad para implementar un escenario. Puede crear credenciales nuevas o utilizar las existentes.

Para obtener más información sobre la configuración de credenciales, consulte Descripción y obtención de las credenciales de AWS.

Para actualizar las credenciales de AWS, realice el siguiente procedimiento:

1. En Unity, en la pestaña del complemento para Unity, elija la pestaña Implementar.
2. En el panel Implementar, elija Credenciales de AWS.
3. Puede crear credenciales de AWS nuevas o utilizar las existentes.
   - Para crear credenciales, elija Crear nuevo perfil de credenciales y, a continuación, especifique el nombre del nuevo perfil, el ID de la clave de acceso de AWS, la clave secreta de AWS y la Región de AWS.
   - Para elegir las credenciales existentes, seleccione Elegir perfil de credenciales existente y, a continuación, seleccione un nombre de perfil y la Región de AWS.
4. En la ventana Actualizar credenciales de AWS, seleccione Actualizar perfil de credenciales.

Actualiza el bootstrap de la cuenta

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

La ubicación de arranque es un bucket de Amazon S3 que se utiliza durante la implementación. Se utiliza para almacenar los recursos del servidor de juegos y otras dependencias. La Región de AWS que elija para el bucket debe ser la misma región que utilizará para la implementación del escenario.

Para obtener más información sobre los buckets de Amazon S3, consulte Creación, configuración y trabajo con buckets de Amazon Simple Storage Service.

Para actualizar la ubicación de arranque de la cuenta, realice el siguiente procedimiento:

1. En Unity, en la pestaña del complemento para Unity, elija la pestaña Implementar.
2. En el panel Implementar, seleccione Actualizar arranque de cuenta.
3. En la ventana de Arranque de cuenta, elija un bucket de Amazon S3 existente o cree un bucket de Amazon S3 nuevo:
   - Para elegir un bucket existente, seleccione Elegir un bucket de Amazon S3 existente y Actualizar para guardar la selección.
   - Seleccione Crear un bucket de Amazon S3 nuevo para crear un nuevo bucket de Amazon Simple Storage Service y, a continuación, elija una política. La política especifica cuándo caducará el bucket de Amazon S3. Elija Crear para crear el bucket.

Implementa un escenario de juego

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

Puedes usar un escenario para probar tu juego con Amazon GameLift. Cada escenario utiliza una plantilla de AWS CloudFormation para crear una pila con los recursos necesarios. La mayoría de los escenarios requieren un servidor de juegos ejecutable y una ruta de compilación. Cuando despliegas el escenario, Amazon GameLift copia los activos del juego en la ubicación de arranque como parte del despliegue.

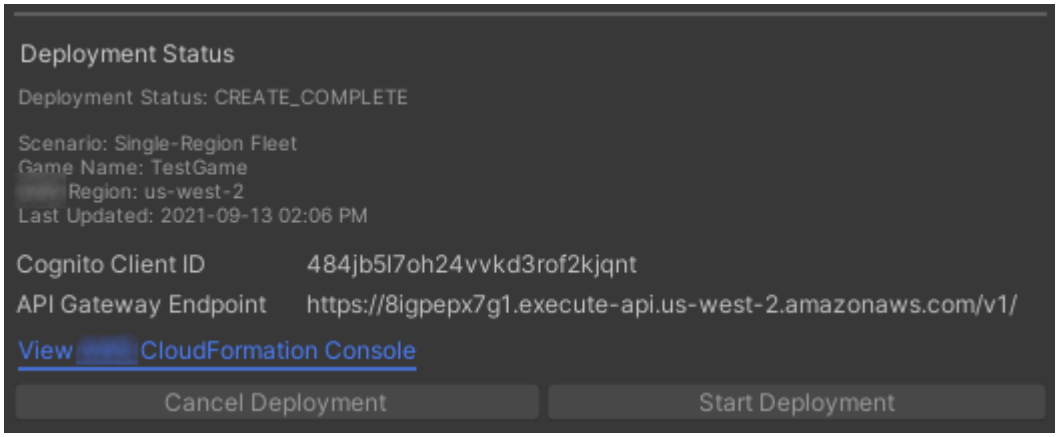
Debe configurar las credenciales de AWS y el arranque de la cuenta de AWS para implementar un escenario.

Para implementar un escenario, realice el siguiente procedimiento:

1. En Unity, en la pestaña del complemento para Unity, elija la pestaña Implementar.
2. En el panel Implementar, seleccione Abrir interfaz de usuario de implementación.
3. En la ventana Implementación, elija un escenario.

4. Especifique un nombre de juego. Deben ser únicos. El nombre del juego forma parte del nombre de la pila de AWS CloudFormation cuando implementa el escenario.
5. Elija la ruta de la carpeta de compilación del servidor de juegos. La ruta de la carpeta de compilación apunta a la carpeta que contiene el archivo ejecutable del servidor y las dependencias.
6. Elija la ruta del archivo .exe de compilación del servidor de juegos. La ruta del archivo ejecutable de compilación apunta al archivo ejecutable del servidor de juegos.
7. Elija Iniciar implementación para comenzar a implementar un escenario. Puede seguir el estado de la actualización en la ventana Implementación, en Estado actual. Los escenarios pueden tardar hasta 30 minutos en implementarse.

8. Cuando el escenario finalice la implementación, el estado actual se actualizará para incluir el ID de cliente de Cognito y el punto de enlace de API Gateway, que podrá copiar y pegar en el juego.



9. Para actualizar la configuración del juego, en el menú de Unity, elija Ir a la configuración de conexión del cliente. Aparecerá la pestaña Inspector en el lado derecho de la pantalla de Unity.
10. Anule la selección de la opción Modo de prueba local.

11. Especifique el punto de conexión de API Gateway y el ID de cliente de Coginito. Elija la misma Región de AWS que utilizó para la implementación del escenario. A continuación, podrá volver a compilar y ejecutar el cliente de juego con los recursos del escenario implementados.

Eliminación de los recursos creados para el escenario

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

Para eliminar los recursos creados para el escenario, elimine la pila de AWS CloudFormation correspondiente.

Para eliminar recursos creados por el escenario, realice el siguiente procedimiento:

1. En la ventana de implementación del GameLift complemento Amazon para Unity, selecciona View AWS CloudFormation Console para abrir la AWS CloudFormation consola.
2. En la consola de AWS CloudFormation, seleccione Pilas y, a continuación, elija la pila que incluye el nombre del juego especificado durante la implementación.
3. Elija Eliminar para borrar la pila. La pila tarda unos minutos en eliminarse. Después de que AWS CloudFormation elimine la pila utilizada en el escenario, su estado cambia a `ROLLBACK_COMPLETE`.

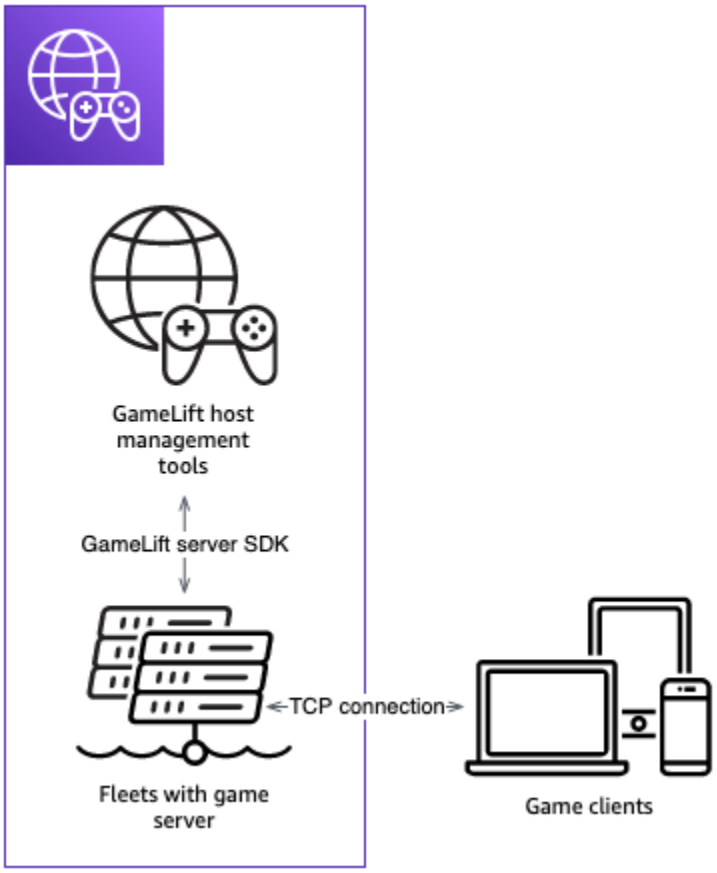
Integra juegos con Amazon GameLift en Unity

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

Integra tu juego de Unity con Amazon GameLift realizando las siguientes tareas:

- Integre Amazon GameLift con un proyecto de servidor de juegos de Unity
- Integre Amazon GameLift con un proyecto de cliente de juegos de Unity

En el siguiente diagrama se muestra un ejemplo de flujo de integración de un juego. En el diagrama, se despliega una flota con el servidor del juego en Amazon GameLift. El cliente del juego se comunica con el servidor del juego, que se comunica con Amazon GameLift.



Importa y ejecuta un juego de muestra

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

El GameLift complemento de Amazon para Unity incluye un juego de muestra que puedes usar para explorar los aspectos básicos de la integración de tu juego con Amazon GameLift. En esta sección, crearás el cliente y el servidor del juego y, a continuación, probarás localmente con Amazon GameLift Local.

Requisitos previos

- Configura un Cuenta de AWS
- Instala y configura el complemento

Compilación y ejecución del servidor de juegos de ejemplo

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

Configure los archivos del servidor de juegos del juego de ejemplo.

1. En Unity, en el menú, selecciona Amazon y GameLift, a continuación, selecciona Importar juego de muestra.
2. En la ventana Importar un juego de muestra, elija Importar para importar el juego, sus recursos y dependencias.
3. Compile el servidor de juegos. En Unity, en el menú, selecciona Amazon y, a continuación GameLift, selecciona Apply Windows Sample Server Build Settings o Apply MacOS Sample Server Build Settings. Después de configurar los ajustes del servidor de juegos, Unity vuelve a compilar los recursos.
4. En Unity, en el menú, elija Archivo y, a continuación, elija Compilación. Elija Compilación del servidor, Compilación y, a continuación, seleccione una carpeta de compilación específica para los archivos del servidor.

   Unity compila el servidor de juegos de ejemplo y coloca el archivo ejecutable y los recursos necesarios en la carpeta de compilación especificada.

Compilación y ejecución del cliente de juegos de ejemplo

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

Configure los archivos del cliente de juegos del juego de ejemplo.

1. En Unity, en el menú, selecciona Amazon y, a continuación GameLift, selecciona Apply Windows Sample Client Build Settings o Apply MacOS Sample Client Build Settings. Una vez configurados los ajustes del cliente de juegos, Unity volverá a compilar los recursos.
2. En Unity, en el menú, seleccione Ir a la configuración del cliente. Aparecerá la pestaña Inspector en el lado derecho de la pantalla de Unity. En la pestaña Amazon GameLift Client Settings, selecciona Modo de prueba local.
3. Compile el cliente de juegos. En Unity, en el menú, elija Archivo. Confirme que la opción Compilación del servidor no esté activa, elija Compilación y, a continuación, seleccione una carpeta de compilación específica para los archivos del cliente.

   Unity compila el cliente de juegos de ejemplo y coloca el archivo ejecutable y los recursos necesarios en la carpeta de compilación del cliente especificada.
4. No ha compilado el servidor y el cliente de juegos. En los siguientes pasos, ejecutas el juego y ves cómo interactúa con Amazon GameLift.

Realización de la prueba del juego de ejemplo de forma local

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

Ejecuta el juego de muestra que has importado con Amazon GameLift Local.

1. Lance el servidor de juegos. En Unity, en la pestaña del complemento para Unity, elija la pestaña Implementar.
2. En el panel Prueba, elija Abrir interfaz de usuario de prueba local.
3. En la ventana Prueba local, especifique una ruta de archivo .exe del servidor de juegos. La ruta debe incluir el nombre del archivo ejecutable. Por ejemplo, `C:/MyGame/GameServer/MyGameServer.exe` .
4. Elija Implementar y ejecutar. El complemento para Unity inicia el servidor del juego y abre una ventana de registro de Amazon GameLift Local. La ventana contiene mensajes de registro, incluidos los mensajes enviados entre el servidor del juego y Amazon GameLift Local.

5. Lance el cliente de juegos. Busque la ubicación de la compilación con el cliente de juegos de ejemplo y elija el archivo ejecutable.
6. En el juego de GameLift muestra de Amazon, introduce un correo electrónico y una contraseña y, a continuación, selecciona Iniciar sesión. El correo electrónico y la contraseña no están validados o no se utilizan.
7. En Amazon GameLift Sample Game, selecciona Start. El cliente de juegos busca una sesión de juego. Si no encuentra una sesión, crea una. A continuación, el cliente de juegos inicia la sesión de juego. Puede ver la actividad del juego en los registros.

Ejemplos de registros del servidor de juegos

```
...
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
```

Ejemplos de registros de 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"
```

Detención de un proceso del servidor

> **Note**
>
> Este tema hace referencia al GameLift complemento Amazon para Unity versión 1.0.0, que usa el SDK de servidor 4.x o anterior.

Cuando termine con el juego de ejemplo, apague el servidor en Unity.

1. En el cliente de juegos, elija Salir o cierre la ventana para detener el cliente de juegos.
2. En Unity, en la ventana Pruebas locales, seleccione Detener o cierre las ventanas del servidor de juegos para detener el servidor.

Integración de juegos con el GameLift complemento de Amazon para Unreal Engine

Los temas de esta sección describen el GameLift complemento de Amazon para Unreal Engine (UE) y cómo usarlo para preparar tu proyecto de juego multijugador para alojarlo en Amazon GameLift. Trabaje completamente en su entorno de desarrollo de UE con los flujos de trabajo guiados del complemento para completar los requisitos básicos de alojamiento con Amazon GameLift.

Amazon GameLift es un servicio totalmente gestionado que permite a los desarrolladores de juegos gestionar y escalar servidores de juegos dedicados para juegos multijugador basados en sesiones. Para obtener más información sobre el GameLift alojamiento de Amazon, consulteCómo funciona Amazon GameLift.

Temas

Acerca del complemento

El complemento añade GameLift las herramientas y funciones de Amazon al editor de la UE. Los flujos de trabajo guiados del complemento para GameLift integrar Amazon en tu proyecto de juego, designar una estación de trabajo como host local para las pruebas e implementar el servidor del juego en el alojamiento GameLift en la nube de Amazon.

Utilice las soluciones de alojamiento prediseñadas del complemento para implementar su juego. Configure una flota de Amazon GameLift Anywhere con su estación de trabajo local como host. Para el alojamiento en la nube, elija entre dos escenarios de implementación comunes que equilibren la latencia de los jugadores, la disponibilidad de las sesiones de juego y el costo de diferentes maneras. Uno de los escenarios incluye un sencillo sistema de FlexMatch emparejamiento y un conjunto de reglas. Utilice estas soluciones para empezar rápidamente con una estructura de alojamiento lista para la producción y, a continuación, realice la optimización y personalización según sea necesario.

El complemento incluye los siguientes componentes:

- Módulos de complementos para el editor de UE. Cuando se instala el complemento, un nuevo botón del menú principal te da acceso a las GameLift funciones de Amazon.
- Bibliotecas C++ para la API de GameLift servicios de Amazon con funcionalidad del lado del cliente.
- Bibliotecas Unreal para el SDK GameLift del servidor Amazon (versión 5).
- Contenido para realizar pruebas, que incluye un mapa del juego inicial y dos mapas de prueba con planos básicos y elementos de la interfaz de usuario para utilizarlos cuando se pruebe la integración de un servidor.
- Configuraciones editables, en forma de plantillas de AWS CloudFormation, que el complemento utiliza cuando se implementa el servidor de juegos como alojamiento.

Flujo de trabajo del complemento

Los siguientes pasos describen un enfoque típico para integrar e implementar un proyecto de juego con el GameLift complemento de Amazon para Unreal Engine. Para completar estos pasos, debe trabajar en el editor de UE y en el código del juego.

1. Crea un perfil de usuario que se vincule a tu AWS cuenta y proporcione las credenciales de acceso de un usuario de cuenta válido con permisos para usar Amazon GameLift.
2. Añade el código de servidor a tu proyecto de juego para establecer la comunicación entre un servidor de juegos en ejecución y el GameLift servicio with Amazon.
3. Añade un código de cliente a tu proyecto de juego que permita a los clientes del juego enviar solicitudes GameLift a Amazon para iniciar nuevas sesiones de juego y, después, conectarse a ellas.
4. Utilice el flujo de trabajo de Anywhere para configurar su estación de trabajo local como alojamiento de Anywhere para su servidor de juegos. Inicie el servidor y el cliente de juegos de forma local a través del complemento, conéctese a una sesión de juego y pruebe la integración.
5. Utilice el flujo de trabajo de alojamiento de EC2 para cargar su servidor de juegos integrado e implementar una solución de alojamiento en la nube. Cuando el servidor de juegos esté listo, inicie el cliente de juegos de forma local a través del complemento, conéctese a una sesión de juego y juegue.

Cuando trabajes en el plugin, crearás y utilizarás AWS recursos. Estas acciones pueden conllevar cargos a la AWS cuenta en uso. Si es la primera vez que lo AWS usas, es posible que estas acciones estén incluidas en la capa AWS gratuita.

Instalación del complemento para Unreal

En esta sección se describen las tareas de instalación iniciales para añadir el complemento a un proyecto de Unreal Engine. La funcionalidad del complemento está disponible cuando tiene el proyecto abierto en el editor de Unreal.

> **Note**
>
> Puedes usar el GameLift plugin Amazon con una versión estándar del editor UE, pero tendrás que usar una versión basada en el código fuente cuando empaques la compilación de tu servidor de juegos.

Antes de comenzar

Esto es lo que necesitas para usar el GameLift plugin de Amazon para Unreal Engine:

- Paquete de lanzamiento GameLift del plugin de Amazon para Unreal Engine. [Sitio de descargas].
- Microsoft Visual Studio 2019 o una versión posterior.
- Una versión original del editor de Unreal Engine. Necesitará una versión original para empaquetar los componentes del servidor para un juego multijugador. Para obtener más información, incluidos los requisitos previos adicionales, consulte la documentación de Unreal Engine:
  - Para acceder al código fuente de Unreal Engine en GitHub GitHub Necesitarás una cuenta de Epic Games.
  - Tutorial Compilación de Unreal Engine a partir del código fuente.
- Un proyecto de juego multijugador con código de juego en C++. Si estás trabajando con un proyecto de Blueprint, consulta la documentación de Unreal sobre cómo generar código fuente en C++ para tu proyecto.

Adición del complemento al proyecto de juego

Completa las siguientes tareas para añadir el plugin a tu proyecto de juego.

Cree el SDK del servidor Amazon GameLift C++

1. Descomprime el paquete de lanzamiento del GameLift plugin Amazon para Unreal Engine para extraer dos archivos zip:

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

   Descomprime estos archivos.
2. Abra la `GameLift-Cpp-ServerSDK-<>` carpeta y, a continuación, complete las siguientes instrucciones para su plataforma: Linux o Microsoft Windows.

Linux

1. Ejecute los comandos siguientes:

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

   Estos comandos crean el `/lib/aws-cpp-sdk-gamelift-server.so` archivo.
2. Copie `/lib/aws-cpp-sdk-gamelift-server.so` al `amazon-gamelift-plugin-unreal/GameLiftPlugin/Source/GameliftServer/ThirdParty/GameLiftServerSDK/Linux/x86_64-unknown-linux-gnu/` directorio.

Microsoft Windows

1. Ejecute los comandos siguientes:

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

   Estos comandos crean los siguientes archivos binarios.

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

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

2. Copie los archivos al `amazon-gamelift-plugin-unreal\GameLiftPlugin\Source\GameliftServer\ThirdParty\GameLiftServerSDK\Win64\` directorio.

Realice los siguientes procedimientos cuando trabaje en los archivos del proyecto del juego:

1. Instale los archivos del complemento.

   a. Busque la carpeta raíz del proyecto del juego, por ejemplo `... > Unreal Projects/[project-name]/`. Si la carpeta de complementos no existe allí, créela.

   b. Ve a la `amazon-gamelift-plugin-unreal` carpeta de la que lo descomprimiste. `amazon-gamelift-plugin-unreal-<>-sdk-<>.zip` Copia la `GameLiftPlugin` carpeta de la `gamelift-plugin-unreal` carpeta a la `Plugins` carpeta del directorio del proyecto del juego.

2. Añada el complemento al archivo **`.uproject`**.

   a. En la carpeta raíz del proyecto de juego, abra el archivo `.uproject`.

   b. Actualiza el archivo para añadir "GameLiftPlugin" y "WebBrowserWidget" a la `Plugins` sección y habilítalos. El siguiente código muestra el `.uproject` archivo actualizado de un juego llamado "MyGame».

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

```
  ]
}
```

3. Cambie la versión del editor de UE para el proyecto.

   Si ha creado un proyecto para una versión del editor y ahora quiere cambiarlo a otra versión (por ejemplo, una versión original), debe actualizar el proyecto.

   En la carpeta raíz del proyecto de juego, seleccione el archivo `.uproject` y elija la opción Cambiar de versión de Unreal Engine. Seleccione una nueva versión del editor.

4. Vuelva a compilar la solución del proyecto con sus actualizaciones.

   a. En la carpeta raíz del proyecto, busque un archivo de solución (`*.sln`). Si no existe ninguno, seleccione el archivo `.uproject` y elija la opción Generar archivos de proyecto de Visual Studio.

   b. Abra el archivo de la solución y cree o reconstruya el proyecto.

5. Verifique que el complemento esté habilitado en el editor de UE.

   > **Note**
   >
   > Si ya tiene el editor abierto, es posible que tenga que reiniciarlo para que reconozca el nuevo complemento.

   a. Abra el proyecto en el editor de UE que haya elegido.

   b. Consulta la barra de herramientas principal del editor para ver el nuevo botón de GameLift menú de Amazon [necesita imagen].

   c. Busca en el navegador de contenido los activos del GameLift plugin de Amazon. Asegúrese de que la configuración de las Opciones de visualización tenga seleccionada la opción Mostrar contenido del complemento.

Configuración de un perfil de usuario de AWS

Después de instalar el complemento, configure un perfil y vincúlelo a un usuario de la cuenta de AWS válido. Puede mantener varios perfiles, pero solo puede tener un perfil activo a la vez. Siempre que trabaje con el complemento, seleccione el perfil que quiera utilizar.

El mantenimiento de varios perfiles le ofrece la posibilidad de cambiar entre diferentes escenarios de alojamiento. Por ejemplo, puede configurar perfiles con las mismas credenciales de AWS, pero con distintas regiones de AWS. También puede configurar perfiles con diferentes cuentas de AWS o con distintos conjuntos de usuarios o permisos.

> **Note**
>
> Si ha instalado la AWS CLI en su estación de trabajo y ya tiene un perfil configurado, el GameLift complemento Amazon puede detectarlo y lo incluirá como un perfil existente. El complemento selecciona automáticamente cualquier perfil nombrado como `[default]`. Puede utilizar un perfil existente o crear uno nuevo.

Para administrar los perfiles de AWS, realice el siguiente procedimiento:

1. En la barra de herramientas principal del editor Unreal, selecciona el GameLift menú Amazon y selecciona Establecer perfiles AWS de usuario. Con esta acción se abrirá la Configuración del proyecto para el complemento. Amplíe la sección Perfiles de usuario de AWS.
2. Si el complemento no detecta un perfil existente, te pedirá que crees uno. Puede crear un perfil nuevo mediante una cuenta de AWS nueva o existente.

   > **Note**
   >
   > Debe usar la consola de administración de AWS para crear una cuenta de AWS nueva y crear o actualizar un usuario con el conjunto de permisos adecuado.

   Al configurar un perfil, deberá proporcionar la siguiente información:

   - Una cuenta de AWS. Si necesita crear una cuenta de AWS nueva, siga las instrucciones para crearla. Consulte Creación de una cuenta de AWS para obtener más información.
   - Un AWS usuario con permisos para usar Amazon GameLift y otros AWS servicios necesarios. Consulte Configura un Cuenta de AWS para obtener instrucciones sobre cómo configurar un usuario AWS Identity and Access Management (IAM) con GameLift permisos de Amazon y acceso programático con credenciales de larga duración.

- Credenciales de su usuario de AWS. Estas credenciales se componen de un ID de clave de acceso de AWS y una clave secreta de AWS. Consulte Obtención de claves de acceso para obtener más información.
- Región de AWS. Se trata de la ubicación geográfica en la que desea crear sus recursos de AWS de alojamiento. Durante el desarrollo, le recomendamos que utilice una región cercana a su ubicación física. Consulte la lista de regiones de AWS admitidas.

3. Si el complemento detecta un perfil existente, no se te pedirá que crees uno. Si quiere actualizar un perfil o crear uno nuevo, seleccione Administrar su perfil.

Para arrancar el perfil, tenga en cuenta la siguiente información:

Todos los perfiles deben estar inicializados para poder utilizarlos con el plugin de Amazon GameLift . Con el arranque, se crea un bucket de Amazon S3 específico para el perfil. Se usa para almacenar configuraciones de proyectos, crear artefactos y otras dependencias. Los buckets no se comparten entre otros perfiles.

El arranque implica la creación de nuevos recursos de AWS, lo que implicar costos.

1. En la barra de herramientas principal del editor Unreal, elige el GameLift icono de Amazon y selecciona Establecer perfiles AWS de usuario. Con esta acción se abrirá la Configuración del proyecto para el complemento. Amplíe la sección Perfiles de usuario de AWS.
2. En la sección Arrancar su perfil, seleccione un perfil de la lista desplegable y compruebe el estado del arranque. Si el estado indica que no existe ningún bucket, pulse el botón Arrancar y crear perfil para crear un bucket de Amazon S3 para el perfil seleccionado.
3. Espere a que el estado del arranque cambie a «Activo». Este proceso puede tardar unos minutos.

Configura tu juego para probarlo con Amazon GameLift Anywhere

En este flujo de trabajo, añades el código de juego de cliente y servidor para las GameLift funciones de Amazon y utilizas el complemento para designar tu estación de trabajo local como host de servidor de juegos de prueba. Cuando haya completado las tareas de integración, utilice el complemento para crear los componentes de cliente y servidor de juegos.

Para iniciar el flujo de trabajo de Amazon GameLift Anywhere:

- En la barra de herramientas principal del editor Unreal, selecciona el GameLift menú Amazon y selecciona Host with Anywhere. Con esta acción se abre la página del complemento Implementar Anywhere, que presenta un proceso de seis pasos para integrar, crear y lanzar los componentes del juego.

Paso 1: Configuración del perfil

Elija el perfil que desee utilizar al seguir este flujo de trabajo. El perfil que seleccione afectará a todos los pasos del flujo de trabajo. Todos los recursos que crees están asociados a la AWS cuenta del perfil y se ubican en la AWS región predeterminada del perfil. Los permisos del usuario del perfil determinan su acceso a AWS los recursos y las acciones.

1. Seleccione un perfil de la lista desplegable de perfiles disponibles. Si aún no tienes un perfil o quieres crear uno nuevo, ve al GameLift menú de Amazon y selecciona Establecer perfiles AWS de usuario.
2. Si el estado de bootstrap no es «Activo», selecciona el perfil de Bootstrap y espera a que el estado cambie a «Activo».

Paso 2: Configuración del código de juego

En este paso, se realizará una serie de actualizaciones en el código del cliente y servidor para añadir la funcionalidad de alojamiento. Si aún no has configurado una versión original del editor Unreal, el complemento proporciona enlaces a las instrucciones y al código fuente.

Con el complemento, podrá aprovechar algunas ventajas a la hora de integrar el código del juego. Puede realizar una integración mínima para configurar la funcionalidad básica de alojamiento. También podrá realizar una integración personalizada más amplia. La información de esta sección describe la opción de integración mínima. Usa los mapas de prueba incluidos con el complemento para añadir la GameLift funcionalidad del cliente Amazon a tu proyecto de juego. Para la integración del servidor, utilice el ejemplo de código proporcionado para actualizar el modo de juego de su proyecto.

Integración del modo de juego del servidor

Añade un código de servidor al juego que permita la comunicación entre el servidor del juego y el GameLift servicio de Amazon. Tu servidor de juegos debe poder responder a las solicitudes de

Amazon GameLift, por ejemplo, para iniciar una nueva sesión de juego, y también informar sobre el estado del servidor de juegos y las conexiones de los jugadores.

1. En el editor de código, abra el archivo de la solución (`.sln`) del proyecto de juego, que normalmente se encuentra en la carpeta raíz del proyecto. Por ejemplo: `GameLiftUnrealApp.sln`.
2. Con la solución abierta, busque el archivo de cabecera del modo de juego del proyecto: archivo `[project-name]GameMode.h`. Por ejemplo: `GameLiftUnrealAppGameMode.h`.
3. Cambie el archivo de encabezado para alinearlo con el siguiente código de ejemplo. Asegúrate de sustituir «GameLiftServer» por el nombre de tu propio proyecto. Estas actualizaciones son específicas del servidor de juegos; le recomendamos que realice una copia de seguridad de los archivos originales del modo de juego para utilizarla con el cliente.

```
// 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. Abra el archivo `[project-name]GameMode.cpp` del archivo de origen relacionado (por ejemplo, `GameLiftUnrealAppGameMode.cpp`). Cambie el código para alinearlo con el siguiente código de ejemplo. Asegúrese de reemplazar «GameLiftUnrealApp» por el nombre de su propio proyecto. Estas actualizaciones son específicas del servidor de juegos; le recomendamos que realice una copia de seguridad del archivo original para utilizarla con el cliente.

   El siguiente código de ejemplo muestra cómo añadir los elementos mínimos necesarios para la integración del servidor con Amazon GameLift:

   - Inicializa un cliente de Amazon GameLift API. La `InitSDK()` llamada con los parámetros del servidor es obligatoria para una flota de Amazon GameLift Anywhere. Cuando se conecta a una flota de Anywhere, el complemento almacena los parámetros del servidor como argumentos de la consola. El código de ejemplo puede acceder a los valores en tiempo de ejecución.
   - Implemente las funciones de devolución de llamada necesarias para responder a las solicitudes del GameLift servicio de Amazon, incluidas `OnStartGameSessionOnProcessTerminate`, yon`HealthCheck`.
   - `ProcessReady()`Llama a un puerto designado para notificar al GameLift servicio de Amazon cuando estés listo para organizar sesiones de juego.

```
// 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
}
```

Integración del mapa de juego del cliente

El mapa inicial del juego contiene una lógica de esquema y elementos de interfaz de usuario que ya incluyen un código básico para solicitar sesiones de juego y utilizar la información de la conexión para conectarse a una sesión de juego. Puede utilizar el mapa tal como está o modificarlo según sea necesario. Utilice el mapa inicial del juego con otros recursos del juego, como el proyecto de plantilla Tercera persona proporcionado por Unreal Engine. Estos recursos están disponibles en el navegador de contenido. Puedes usarlos para probar los flujos de trabajo de despliegue del plugin o como guía para crear un servicio de backend personalizado para tu juego.

El mapa inicial tiene las siguientes características:

- Incluye la lógica tanto para una flota de Anywhere como para una flota de EC2 administrada. Al gestionar su cliente, puede elegir conectarse a cualquiera de las dos flotas.
- La funcionalidad del cliente incluye buscar una sesión de juego SearchGameSessions (()), crear una nueva sesión de juego (CreateGameSession()) y unirse a una sesión de juego directamente.
- Obtiene un ID de jugador único del grupo de usuarios de Amazon Cognito del proyecto (forma parte de una solución de Anywhere implementada).

Para usar el mapa del juego inicial, realice el siguiente procedimiento:

1. En el editor de UE, abra la página Configuración, mapas y modos del proyecto y expanda la sección Mapas predeterminados.
2. En el mapa de inicio del editor, selecciona StartupMap "» en la lista desplegable. Puede que necesite buscar el archivo, que se encuentra en `... > Unreal Projects/[project-name]/Plugins/Amazon GameLift Plugin Content/Maps`.
3. En el mapa predeterminado del juego, selecciona el mismo "StartupMap" en la lista desplegable.
4. Para el mapa predeterminado del servidor, selecciona "ThirdPersonMap». Este es un mapa predeterminado incluido en el proyecto de juego. Este mapa está diseñado para dos jugadores del juego.
5. Abra el panel de detalles del mapa predeterminado del servidor. Defina GameMode la anulación como «Ninguna».
6. Expanda la sección Modos predeterminados y establezca Modo de juego del servidor predeterminado global en el modo de juego que actualizó para la integración del servidor.

Una vez que hayas realizado estos cambios en tu proyecto, estarás listo para crear los componentes del juego.

Compilación de los componentes del juego

1. Creación de nuevos archivos de destino de servidor y cliente

    a. En la carpeta de proyectos del juego, diríjase a la carpeta Fuente y busque los archivos `Target.cs`.

    b. Copie el archivo `[project-name]Editor.Target.cs` en dos archivos nuevos denominados `[project-name]Client.Target.cs` y`[project-name]Server.Target.cs`.

    c. Edite cada uno de los archivos nuevos para actualizar el nombre de la clase y los valores del tipo de destino, como aparece a continuación:

```
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. Actualice el archivo `.Build.cs`.

   a. Abra el archivo `.Build.cs` para su proyecto. Este archivo se encuentra en `UnrealProjects/[project name]/Source/[project name]/[project name].Build.cs`.

   b. Actualice la clase `ModuleRules` como aparece en el siguiente ejemplo de código.

```
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. Vuelva a compilar la solución del proyecto de juego.
4. Abra el proyecto de juego en una versión original del editor de Unreal Engine.
5. Realice el siguiente procedimiento tanto para el cliente como para el servidor:

   a. Elija un destino. Diríjase a Plataformas, Windows y seleccione una de las siguientes opciones:

      - Servidor: `[your-application-name]Server`
      - Cliente: `[your-application-name]Client`

   b. Iniciar la compilación. Diríjase a Plataforma, Windows, Proyecto de paquete.

Cada proceso de empaquetado genera un archivo ejecutable: `[your-application-name]Client.exe` o `[your-application-name]Server.exe`.

En el complemento, establezca las rutas a los archivos ejecutables de compilación del cliente y del servidor en su estación de trabajo local.

Paso 3: Conexión a una flota de Anywhere

En este paso, tendrá que designar la flota de Anywhere que desee utilizar. Una flota de Anywhere define un conjunto de recursos informáticos, que se pueden ubicar en cualquier lugar para el alojamiento de servidores de juegos.

- Si la AWS cuenta que utilizas actualmente ya tiene flotas de Anywhere, abre el campo desplegable del nombre de la flota y elige una flota. Este menú desplegable solo muestra las flotas de Anywhere de la región de AWS para el perfil de usuario activo actualmente.
- Si no hay ninguna flota existente, o si desea crear una nueva, seleccione Crear nueva flota Anywhere e introduzca un nombre para la flota.

Una vez que hayas elegido una flota de Anywhere para tu proyecto, Amazon GameLift verifica que el estado de la flota esté activo y muestra el identificador de la flota. Puedes hacer un seguimiento del progreso de esta solicitud en el registro de resultados del editor de Unreal.

Paso 4: Registro de su estación de trabajo

En este paso, se registrará su estación de trabajo local como recurso informático en la nueva flota de Anywhere.

1. Especifique un nombre de equipo para la máquina local. Si añade más de un recurso informático a la flota, los nombres deben ser únicos.
2. Proporcione una dirección IP para su máquina local. Este campo está predeterminado en la dirección IP pública de tu máquina. También puedes usar localhost (127.0.0.1) siempre que ejecutes el cliente y el servidor del juego en la misma máquina.
3. Elija Registrar computación. Puedes hacer un seguimiento del progreso de esta solicitud en el registro de resultados del editor de Unreal.

En respuesta a esta acción, Amazon GameLift comprueba que puede conectarse al equipo y devuelve información sobre el equipo recién registrado. También crea los argumentos de consola que los ejecutables de tus juegos necesitan al inicializar la comunicación con el servicio de Amazon GameLift .

Paso 5: Generación del token de autenticación

Los procesos del servidor de juegos que se ejecutan en tu ordenador de Anywhere necesitan un token de autenticación para realizar llamadas al GameLift servicio. El complemento genera y almacena automáticamente un token de autenticación para la flota de Anywhere cada vez que inicia el servidor de juegos desde el complemento. El valor del token de autenticación se almacena como un argumento de línea de comandos, que el código del servidor puede recuperar en tiempo de ejecución.

No tiene que realizar ninguna acción en este paso.

Paso 6: Lanzamiento del juego

En este punto, has completado todas las tareas necesarias para lanzar y jugar tu juego multijugador en una estación de trabajo local con Amazon GameLift.

1. Lance el servidor de juegos. El servidor del juego notificará a Amazon GameLift cuando esté listo para organizar sesiones de juego.
2. Lance el cliente de juegos y utilice la nueva funcionalidad para iniciar una nueva sesión de juego. Esta solicitud se envía a Amazon a GameLift través del nuevo servicio de backend. En respuesta GameLift, Amazon llama al servidor del juego, que se ejecuta en tu máquina local, para iniciar una nueva sesión de juego. Cuando la sesión de juego esté lista para aceptar jugadores, Amazon GameLift proporcionará la información de conexión para que el cliente del juego se una a la sesión de juego.

Implementación del juego en un alojamiento en la nube con flotas de EC2 administradas

En este flujo de trabajo, utilizas el complemento para modificar tu juego y alojarlo en recursos informáticos basados en la nube gestionados por Amazon GameLift. Añades el código del juego de cliente y servidor para la GameLift funcionalidad de Amazon y, a continuación, subes la versión de tu servidor al GameLift servicio de Amazon para desplegarla en los recursos basados en la nube. Cuando se complete este flujo de trabajo, dispondrás de un cliente de juego en funcionamiento que podrá conectarse a tus servidores de juegos en la nube.

Para iniciar el flujo de trabajo de Amazon EC2 GameLift gestionado por Amazon:

- En la barra de herramientas principal del editor Unreal, elija el GameLift menú Amazon y seleccione Host with Managed EC2. Con esta acción, se abrirá la página del complemento Implementar flota de Amazon EC2, que presenta un proceso de seis pasos para integrar, compilar y lanzar los componentes del juego.

Paso 1: Configuración del perfil

Elija el perfil que desee utilizar al seguir este flujo de trabajo. El perfil que seleccione afectará a todos los pasos del flujo de trabajo. Todos los recursos que cree están asociados a la AWS cuenta del perfil y se ubican en la región predeterminada AWS del perfil. Los permisos del usuario del perfil determinan su acceso a AWS los recursos y las acciones.

1. Seleccione un perfil de la lista desplegable de perfiles disponibles. Si aún no tienes un perfil o quieres crear uno nuevo, ve al GameLift menú de Amazon y selecciona Establecer perfiles AWS de usuario.
2. Si el estado de bootstrap no es «Activo», selecciona el perfil de Bootstrap y espera a que el estado cambie a «Activo».

Paso 2: Configuración del código de juego

En este paso, se realizará una serie de actualizaciones en el código del cliente y servidor para añadir la funcionalidad de alojamiento. Si aún no has configurado una versión original del editor Unreal, el complemento proporciona enlaces a las instrucciones y al código fuente.

Si ha integrado el juego para usarlo con una flota de Anywhere, no es necesario realizar ningún cambio en el código de juego. Si utiliza el mapa del juego inicial, también funciona con las implementaciones de EC2.

- Configuración del código del juego (Anywhere)
- Compilación de los componentes del juego

Tras crear tu servidor de juegos, completa las siguientes tareas para prepararlo para subirlo a Amazon GameLift.

Para empaquetar la compilación del servidor para la implementación en la nube, realice el siguiente procedimiento:

En la carpeta `WindowsServer`, donde el editor de Unreal empaqueta los archivos de compilación del servidor de forma predeterminada, realice el siguiente procedimiento:

1. Copie el script de instalación, incluido en la descarga del complemento, en la raíz de la carpeta `WindowsServer`. Busque el archivo `[project-name]/Plugins/Resources/CloudFormation/extra_server_resources/install.bat`. Amazon GameLift utiliza este archivo para instalar la compilación del servidor en cada recurso de alojamiento de EC2.
2. Copie el archivo `VC_redist.x64.exe`, incluido en la instalación de Visual Studio, en la raíz de la carpeta `WindowsServer`. Por lo general, este archivo se encuentra en `C:/Program Files (x86)/Microsoft Visual Studio/2019/Professional/VC/Redist/MSVC/v142`.

3. Copie los archivos DLL de OpenSSL para la compilación del servidor de juegos en la carpeta `WindowsServer/MyGame/Binaries/Win64`. Asegúrese de que los archivos DLL sean para la misma versión que se utiliza en la compilación del servidor. Copie los siguientes archivos:

   - `libssl-3-x64.dll`
   - `libcrypto-3-x64.dll`

Paso 3: Selección del escenario de implementación

En este paso, tendrá que elegir la solución de alojamiento de juegos que desee implementar en ese momento. Puede disponer de varias implementaciones del juego mediante cualquiera de los escenarios.

- Flota de una sola región: despliega el servidor de juegos en una sola flota de recursos de alojamiento en la región predeterminada del perfil activo. AWS Este escenario es un buen punto de partida para probar la integración del servidor con AWS y la configuración de compilación del servidor. Permite implementar los siguientes recursos:
  - La flota de AWS (bajo demanda) con la compilación del servidor de juegos instalada y en ejecución.
  - Grupo de usuarios y cliente de Amazon Cognito para permitir a los jugadores autenticarse e iniciar un juego.
  - Autorizador de la puerta de enlace de API que vincula el grupo de usuarios con las API.
  - WebACI para limitar las llamadas excesivas de los jugadores a la puerta de enlace de la API.
  - Puerta de enlace de la API + función de Lambda para que los jugadores soliciten una ranura de juego. Esta función llama a `CreateGameSession()` si no hay ninguna disponible.
  - Puerta de enlace de la API + función de Lambda para que los jugadores obtengan información de la conexión para su solicitud de juego.
- FlexMatch flota: despliega tu servidor de juego en un conjunto de flotas y configura un FlexMatch emparejador con reglas para crear partidas de jugadores. En este escenario, se utiliza un alojamiento Spot de bajo coste con una estructura de varias flotas y ubicaciones para garantizar una disponibilidad duradera. Este enfoque resulta útil cuando estás listo para empezar a diseñar un componente de emparejamiento para tu solución de alojamiento. En este escenario, crearás los recursos básicos para esta solución, que podrás personalizar más adelante según sea necesario. Permite implementar los siguientes recursos:

- FlexMatch Configuración y reglas de emparejamiento establecidas para aceptar las solicitudes de los jugadores y formar partidas.
- Tres flotas de AWS con la compilación del servidor de juegos instalada y en ejecución en varios lugares. Incluye dos flotas de spot y una flota bajo demanda como respaldo.
- Cola de ubicación de sesión de juego de AWS que responde a las solicitudes de emparejamientos propuestos mediante la búsqueda del mejor recurso de alojamiento posible (en función de la viabilidad, el costo, la latencia de los jugadores, etc.) y el inicio de una sesión de juego.
- Grupo de usuarios y cliente de Amazon Cognito para permitir a los jugadores autenticarse e iniciar un juego.
- Autorizador de la puerta de enlace de API que vincula el grupo de usuarios con las API.
- WebACI para limitar las llamadas excesivas de los jugadores a la puerta de enlace de la API.
- Puerta de enlace de la API + función de Lambda para que los jugadores soliciten una ranura de juego. Esta función llama a `StartMatchmaking()`.
- Puerta de enlace de la API + función de Lambda para que los jugadores obtengan información de la conexión para su solicitud de juego.
- Tablas de Amazon DynamoDB para almacenar tickets de emparejamiento para jugadores e información sobre las sesiones de juego.
- Tema de SNS más función Lambda para GameSessionQueue gestionar eventos.

Paso 4: Configuración de los parámetros del juego

En este paso, deberá describir el juego que quiera subir a AWS.

- Nombre de la compilación del servidor: proporciona un nombre descriptivo para la compilación del servidor de juegos. AWSusa este nombre para hacer referencia a la copia de la versión del servidor que se carga y se usa para las implementaciones.
- SO de compilación del servidor: especifique el sistema operativo para el que se ha diseñado el servidor. Esto indica a AWS qué tipo de recursos informáticos utilizar para alojar el juego.
- Carpeta del servidor de juegos: permite identifica la ruta a la carpeta de compilación del servidor local.
- Compilación del servidor de juegos: permite identificar la ruta al archivo ejecutable del servidor de juegos.
- Ruta del cliente de juego: permite identificar la ruta al archivo ejecutable del cliente de juego.

- Resultado de la configuración del cliente: este campo debe apuntar a una carpeta de la compilación del cliente que contenga la configuración de AWS. Búsquelo en la siguiente ubicación: `[client-build]/[project-name]/Content/CloudFormation`.

Paso 5: Implementación del escenario

En este paso deberá implementar el juego en una solución de alojamiento en la nube en función del escenario de implementación que elija. Este proceso puede tardar hasta 40 minutos y, además, AWS valida la compilación del servidor, realiza un aprovisionamiento de los recursos de alojamiento, instala el servidor de juegos y lanza los procesos del servidor y los prepara para alojar sesiones de juego.

Para iniciar la implementación, elija Implementar CloudFormation. Puede realizar el seguimiento del estado del alojamiento de su juego aquí. Para obtener información más detallada, puede iniciar sesión en la consola de administración de AWS para AWS y ver las notificaciones de eventos. Asegúrese de iniciar sesión con la misma cuenta, usuario y región de AWS que el perfil de usuario activo del complemento.

Cuando se complete la implementación, tendrá el servidor de juegos instalado en una instancia de EC2 de AWS. Hay al menos un proceso del servidor en ejecución y listo para iniciar una sesión de juego.

Paso 6: Lanzamiento del cliente

En este punto, has completado todas las tareas necesarias para lanzar y jugar a tu juego multijugador alojado en Amazon GameLift. Para jugar al juego, inicie una instancia del cliente de juego.

Si ha implementado el escenario de flota única, puede abrir una instancia de cliente única con un jugador, especificar el mapa del servidor y desplazarse. Abra instancias adicionales del cliente de juego para añadir un segundo jugador al mismo mapa de juego del servidor.

Si has implementado el FlexMatch escenario, la solución espera a que al menos dos clientes estén en cola para situarlos en la sesión de juego antes de que los jugadores puedan entrar en el mapa del servidor.

Obtención de datos de la flota para una instancia de Amazon GameLift

Hay algunas situaciones en las que la compilación de un juego personalizado o el script de Realtime Servers pueden requerir información sobre la flota de Amazon GameLift. Por ejemplo, la compilación o el script del juego pueden incluir código para realizar las siguientes acciones:

- Supervisar la actividad en función de los datos de la flota.
- Reunir las métricas para realizar un seguimiento de la actividad según los datos de la flota. (Muchos juegos utilizan estos datos para las actividades de LiveOps).
- Proporcionar datos relevantes a los servicios de juegos personalizados, por ejemplo, para el emparejamiento, la escalación de la capacidad adicional o la realización de pruebas.

La información de la flota se encuentra disponible como un archivo JSON en cada instancia en las siguientes ubicaciones:

- Windows: `C:\GameMetadata\gamelift-metadata.json`
- Linux: `/local/gamemetadata/gamelift-metadata.json`

El archivo `gamelift-metadata.json` incluye los atributos de un recurso de flota de Amazon GameLift.

Ejemplo de archivo 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"
}
```

Adición del emparejamiento de FlexMatch

Utilice Amazon GameLift FlexMatch para añadir la funcionalidad de emparejamiento de jugadores a sus juegos alojados en Amazon GameLift. Puede utilizar FlexMatch con servidores de juegos personalizados o Realtime Servers.

FlexMatch combina el servicio de emparejamiento con un motor de reglas personalizable. Diseñe la forma de emparejar a los jugadores en función de los atributos de los jugadores y los modos de juego que mejor se adapten a su juego. FlexMatch administra los aspectos básicos de evaluar a los jugadores que buscan un juego, formar emparejamientos con uno o más equipos e iniciar sesiones de juego para alojar los emparejamientos.

Para utilizar el servicio de FlexMatch completo, debe tener los recursos de alojamiento configurados con colas. Amazon GameLift utiliza colas para localizar las mejores ubicaciones de alojamiento posibles para juegos en varias regiones y tipos informáticos. En concreto, las colas de Amazon GameLift pueden usar los datos de latencia, cuando los proporcionan los clientes del juego, para ubicar las sesiones de juego de forma que los jugadores experimenten la latencia más baja posible al jugar.

Para obtener más información sobre FlexMatch, incluida la ayuda detallada para integrar el emparejamiento en sus juegos, consulte estos temas de la Guía para desarrolladores de FlexMatch de Amazon GameLift FlexMatch:

- Cómo funciona Amazon GameLift FlexMatch
- Guía de integración de FlexMatch

Administrar el alojamiento con GameLift contenedores de Amazon

Esta documentación corresponde a una función que se encuentra en una versión preliminar pública. Está sujeta a cambios.

Amazon GameLift ofrece un servicio completo de alojamiento en la nube para admitir soluciones en contenedores para el alojamiento de servidores de juegos. Con las flotas de GameLift contenedores de Amazon, puedes aprovechar las ventajas de los contenedores, como la portabilidad, la agilidad y la tolerancia a fallos.

Características principales

Las siguientes funciones están disponibles en las flotas de GameLift contenedores de Amazon.

- Desarrolle una arquitectura de contenedores personalizada con contenedores livianos para ejecutar el software de su servidor de juegos en los recursos de GameLift alojamiento de Amazon.
- Incluye Amazon GameLift Agent para gestionar el ciclo de vida de los procesos del servidor de juegos dentro de tus contenedores. El agente informático sigue tus instrucciones sobre cuándo y cómo iniciar los procesos del servidor y cuántos mantener para el alojamiento de las sesiones de juego.
- Personaliza los recursos proporcionados por Amazon GameLift para crear imágenes de contenedores con tu aplicación de servidor de juegos. Usa el dockerfile proporcionado para crear una imagen de contenedor basada en Linux. Guarde las imágenes de sus flotas de contenedores en un repositorio privado de Amazon Elastic Container Registry (Amazon ECR).
- Ofrezca experiencias de baja latencia a los jugadores desplegando recursos de flota de contenedores en cualquier zona local Región de AWS o en cualquier zona GameLift compatible con Amazon. Cree flotas de contenedores con múltiples ubicaciones para agilizar la gestión de la flota. Consulte Ubicaciones GameLift de alojamiento de Amazon.
- Prueba tus soluciones de alojamiento de juegos en contenedores con una flota de Amazon GameLift Anywhere. Usa Anywhere para probar localmente el desarrollo de tu solución, incluida la integración de Amazon GameLift SDK y las configuraciones de las imágenes del contenedor.
- Realice un seguimiento del rendimiento del alojamiento de juegos con métricas de rendimiento específicas de los contenedores. Supervise el estado de los recursos de su flota mediante métricas de hardware.

- Usa las herramientas de ubicación de las sesiones de GameLift juego de Amazon, incluidas las colas y el FlexMatch matchmaking, para unir a los jugadores con las mejores sesiones de juego posibles alojadas en tus flotas de contenedores.
- Gestiona los recursos de la flota de contenedores mediante AWS CloudFormation plantillas para Amazon GameLift.

Uso de flotas de contenedores durante la versión preliminar pública

La nueva función de flotas de contenedores se encuentra actualmente en versión preliminar pública. Durante esta fase, se admiten las siguientes GameLift funciones de Amazon:

- Usa flotas de contenedores para alojar servidores de juegos diseñados para Linux. Las flotas de contenedores utilizan `Amazon_Linux_2023` y admiten imágenes de contenedores de Linux. Los contenedores de Windows no son compatibles.
- Integre los proyectos de servidores de juegos únicamente con la versión 5+ GameLift del SDK para servidores de Amazon. No se admiten las versiones anteriores.
- Utilice cualquiera de los tipos de instancias bajo demanda de Amazon EC2 compatibles con Amazon GameLift . Las flotas puntuales no son compatibles en este momento.

Cómo funcionan los contenedores en Amazon GameLift

Esta documentación corresponde a una función que se encuentra en una versión preliminar pública. Está sujeta a cambios.

Las flotas de GameLift contenedores de Amazon están diseñadas para ofrecer flexibilidad a la hora de implementar y escalar las aplicaciones contenerizadas. Utiliza Amazon Elastic Container Service (Amazon ECS) para gestionar el despliegue y la ejecución de tareas para sus flotas de Amazon GameLift . En este tema se describen los elementos estructurales básicos para el funcionamiento de los contenedores en una flota GameLift gestionada por Amazon, se ilustran las arquitecturas comunes y se describen algunos conceptos básicos.

Componentes de la flota de contenedores

Flota

Una flota de contenedores es un conjunto de instancias de Amazon EC2, gestionadas por Amazon GameLift, que ejecutan tus servidores de juegos en contenedores. Al crear una flota,

se configura la forma en que se implementan la arquitectura de contenedores y el software del servidor de juegos en cada instancia de la flota. Puede desplegar una flota de contenedores en una Región de AWS o varias ubicaciones geográficas. Puedes usar las herramientas de escalado GameLift manual o automático de Amazon para escalar la capacidad de una flota de contenedores para albergar sesiones de juego y jugadores.

instancia

Una instancia Amazon EC2 es el servidor virtual que proporciona capacidad informática para el alojamiento de juegos. Con Amazon GameLift, puedes elegir entre una variedad de tipos de instancias. Cada tipo de instancia ofrece una combinación diferente de CPU, memoria, almacenamiento y capacidad de red.

Cuando creas una flota de contenedores, Amazon GameLift despliega instancias en función del tipo de instancia que elijas y de la configuración de tu flota. Cada instancia de flota desplegada es idéntica y ejecuta el software de servidor de juegos en contenedores de la misma manera. El número de instancias de una flota determina el tamaño de la flota y la capacidad de alojamiento de juegos.

Grupo de contenedores

Amazon GameLift utiliza el concepto de grupo de contenedores para describir y gestionar un conjunto de contenedores. Un grupo de contenedores es similar a una «tarea» o «módulo» de contenedores. Dentro de cada grupo de contenedores, puedes definir cómo comparten los contenedores los recursos de CPU y memoria disponibles. También puedes configurar las dependencias entre los contenedores y gestionar el ciclo de vida del grupo de contenedores.

Los grupos de contenedores se pueden replicar en cada instancia de la flota para optimizar el uso de los recursos. Para gestionar la replicación, defina la estrategia de programación de un grupo de contenedores, de la siguiente manera:

- Los grupos de contenedores de réplicas administran los contenedores en los que se ejecuta la aplicación de servidor de juegos y el software de soporte. Todas las flotas de contenedores deben definir un grupo de contenedores de réplicas. Un grupo de réplicas puede tener varias copias en cada instancia de flota, según los requisitos del grupo de contenedores y los recursos del tipo de instancia que se utilice. Todos los contenedores del grupo de réplicas se escalan automáticamente en una instancia.
- Los grupos de contenedores Daemon, que son opcionales, pueden resultar útiles para ejecutar servicios en segundo plano o programas de utilidades, por ejemplo, para la supervisión. El software de tu servidor de juegos no depende directamente de los procesos de un grupo de demonios. Los grupos de contenedores de demonios no se replican: cada instancia de la flota

tiene como máximo una copia del grupo de demonios. Esto significa que los contenedores de un grupo de demonios no escalan en una instancia de flota junto con los contenedores de un grupo de réplicas.

Una flota de contenedores debe tener un grupo de contenedores de réplicas y, de forma opcional, puede tener un grupo de demonios.

Contenedor

El contenedor es el elemento más básico de una arquitectura basada en contenedores. Consiste en una imagen de contenedor con ejecutables de software y archivos dependientes. Cuando defines un contenedor para usarlo con Amazon GameLift, configuras cómo se ejecuta el software en el contenedor.

Cada grupo de contenedores de una flota de contenedores debe tener un contenedor designado como «esencial». Un contenedor esencial impulsa el ciclo de vida de un grupo de contenedores. Si el contenedor esencial falla, se reinicia todo el grupo de contenedores.

Los tipos de contenedores incluyen:

- El contenedor de réplicas esencial incluye todo lo que necesitas para ejecutar los procesos del servidor de juegos y alojar sesiones de juego para los jugadores. Incluye la versión del servidor de juegos, que está integrada con el SDK GameLift del servidor de Amazon, y el software dependiente. También incluye Amazon GameLift Agent, que gestiona el ciclo de vida de los procesos del servidor de juegos. El grupo de réplicas de contenedores de una flota tiene exactamente un contenedor de réplica esencial.
- Los contenedores de réplica no esenciales, también denominados contenedores «sidecar», ejecutan software compatible con la aplicación de servidor de juegos. El uso de un contenedor sidecar te permite ejecutar y escalar el software de soporte junto con tus servidores de juegos, pero administrarlo como contenedores independientes. Si este tipo de contenedor falla, solo se reinicia el contenedor en sí; el grupo de contenedores no se ve afectado.
- Los contenedores Daemon ejecutan un servicio daemon para gestionar los procesos en segundo plano. Un uso común de un contenedor daemon es ejecutar un agente de Amazon CloudWatch (CloudWatch) para recopilar métricas, registros y trazas de tus contenedores. Los contenedores daemon pueden ser esenciales o no, según el momento en que un fallo en un contenedor provoque el reinicio de un grupo de contenedores.

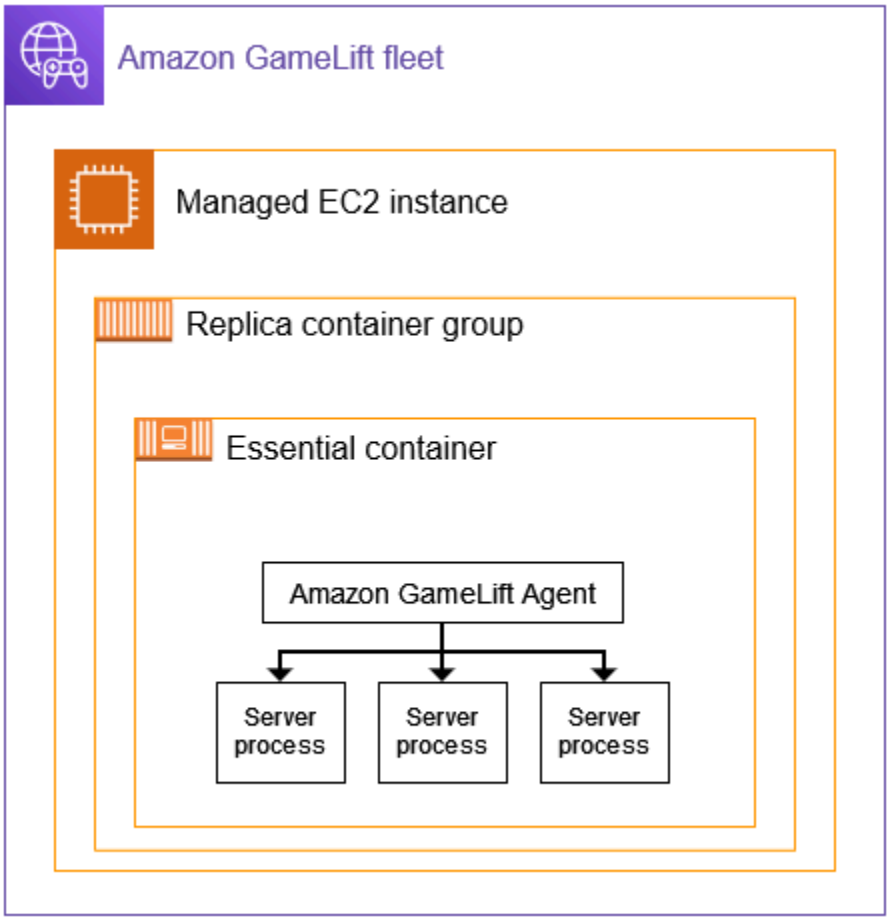
Cálculo

Un ordenador es un recurso de alojamiento de flotas registrado en el GameLift servicio de Amazon y capaz de comunicarse con el servicio. En una flota de contenedores, un cómputo

es un contenedor con un proceso que gestiona el proceso de registro del cómputo. En el contenedor de réplica esencial de una flota de contenedores, el Amazon GameLift Agent registra automáticamente este contenedor como un cómputo.

Arquitecturas comunes

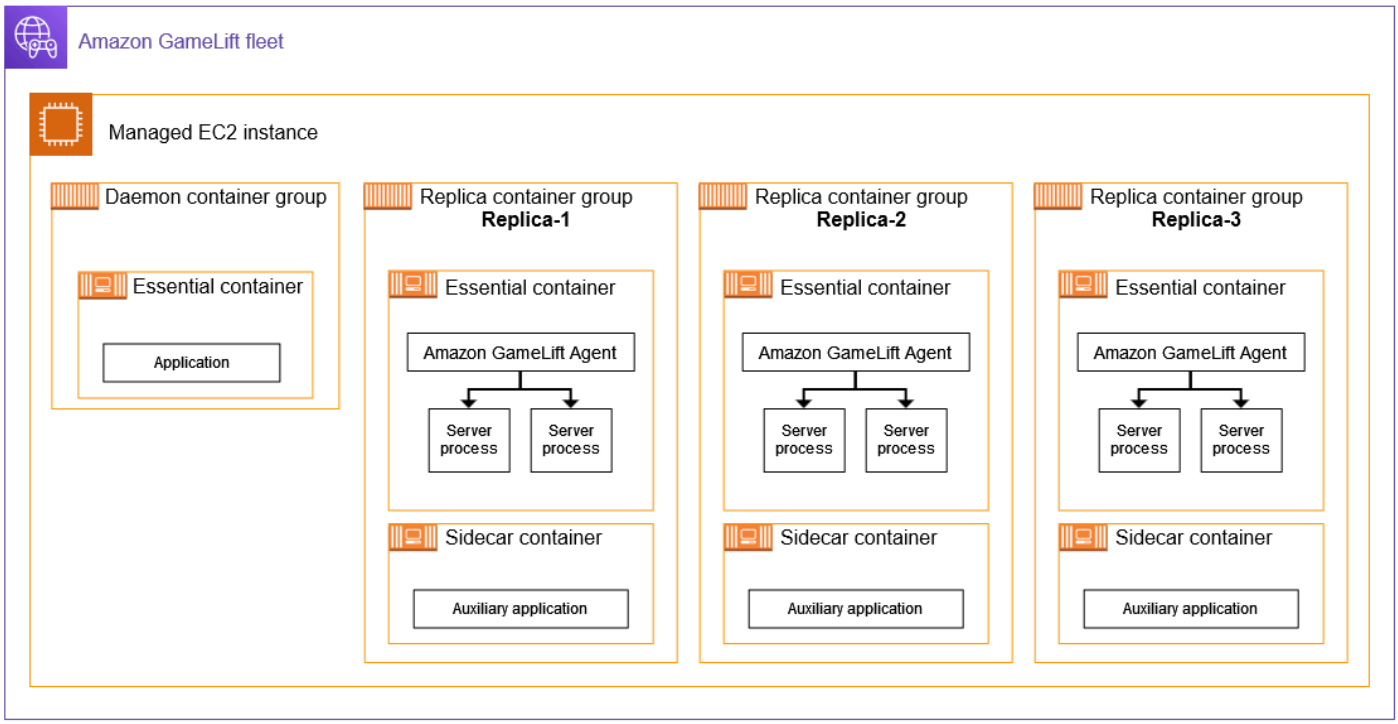
El siguiente diagrama ilustra la estructura de flota de contenedores más simple. En esta estructura, cada instancia de la flota mantiene una copia del grupo de réplicas de contenedores. El grupo de contenedores tiene un único contenedor esencial que ejecuta el Amazon GameLift Agent, la aplicación del servidor del juego y todo el software de soporte para alojar las sesiones de juego. El agente implementa instrucciones específicas de la flota para ejecutar tres procesos del servidor de forma simultánea. Como hay un grupo de contenedores de réplicas por instancia, cada instancia de flota ejecuta tres procesos de servidor simultáneamente.



Este segundo ejemplo ilustra un diseño de flota de contenedores más complejo. En este ejemplo, la flota tiene un grupo de contenedores de réplica con varios contenedores y un grupo de

contenedores daemon con un contenedor. La configuración de la flota incluye tres copias del grupo de contenedores de réplicas en cada instancia de flota. El grupo de contenedores daemon nunca se replica.

Cada conjunto de contenedores de grupos de réplicas que contiene tiene tres copias en cada instancia. En cada contenedor de réplicas esencial, el agente recibe instrucciones de ejecutar dos procesos de servidor simultáneamente. Como resultado, cada instancia de flota ejecuta seis procesos de servidor simultáneamente (dos procesos en cada uno de los tres contenedores de réplica esenciales).



Conceptos clave

En esta sección se resume cómo Amazon GameLift implementa algunos conceptos básicos de contenedores. Para obtener instrucciones sobre cómo trabajar con flotas de contenedores, consulta los temas relevantes de esta guía.

Embalaje para grupos de contenedores

Al desarrollar la estructura de contenedores para su despliegue en una flota de contenedores, un objetivo común es optimizar el uso de la potencia informática disponible. Para lograr este objetivo, busca el mayor número de grupos de réplicas de contenedores que puedas colocar en una instancia de flota sin que ello afecte al rendimiento del servidor de juegos.

Amazon GameLift puede ayudarte a hacerlo. Calcula el número máximo de grupos de réplicas por instancia, en función de la siguiente información:

- El tipo de instancia de la flota y los recursos de CPU y memoria disponibles.
- Los requisitos de CPU y memoria que configuraste para todos los contenedores de tu grupo de réplicas.

  Los requisitos de CPU y memoria que estableces para todos los contenedores de un grupo de demonios, si lo hay.
- Recursos reservados para administrar los contenedores y otras aplicaciones críticas en cada instancia.

Al crear una flota de contenedores, puede elegir usar el número máximo calculado o puede anular el número calculado especificando el número deseado. Como práctica recomendada, experimente con el software de servidor de juegos en contenedores para determinar con precisión los requisitos de recursos. Usa estos datos para encontrar una estrategia de empaquetado óptima para el rendimiento del servidor de juegos.

Servidores de juegos y Amazon GameLift Agent

Cuando construyes tu contenedor de réplicas esencial, empaquetas el software del servidor de juegos y el Amazon GameLift Agent en la misma imagen de contenedor. Este agente informático controla el ciclo de vida de los servidores de juegos del contenedor. En cada grupo de contenedores de réplicas, el contenedor de réplicas esencial ejecuta el agente y todos los procesos del servidor de juegos.

El Amazon GameLift Agent ejecuta las instrucciones en la configuración de tiempo de ejecución de la flota de contenedores. La configuración del tiempo de ejecución identifica (1) el ejecutable que se va a empezar a ejecutar, (2) un conjunto opcional de parámetros de lanzamiento y (3) el número de procesos que se van a ejecutar simultáneamente. Una configuración en tiempo de ejecución puede tener instrucciones para varios ejecutables diferentes. Debe haber al menos una instrucción para el ejecutable del servidor de juegos. Por ejemplo, una configuración de tiempo de ejecución podría indicar al agente que mantenga 10 procesos del servidor de juegos ejecutables para su uso en producción, 1 proceso del mismo ejecutable con parámetros de lanzamiento especiales para las pruebas y 1 proceso para una utilidad de registro.

Puedes modificar la configuración de tiempo de ejecución de una flota en cualquier momento. El GameLift agente de Amazon solicita periódicamente actualizaciones al servicio. Cuando hay

disponible una configuración de tiempo de ejecución actualizada, el agente la recibe y comienza a implementar las instrucciones. Las acciones pueden incluir agregar o cerrar procesos del servidor.

El Amazon GameLift Agent es una versión de código abierto del agente informático que Amazon GameLift utiliza para las flotas de EC2 gestionadas. Esta guía proporciona instrucciones sobre cómo crear el agente a partir del código fuente y convertirlo en una imagen de contenedor. El agente se encarga de las siguientes tareas:

Administración de procesos del servidor:

- Inicie, apague y reemplace los procesos del servidor en función de la configuración del tiempo de ejecución.
- Cierre los procesos del servidor cuando no se activen a tiempo.
- Informe a Amazon GameLift cuando finalice un proceso de servidor.
- Emite eventos de flota para los procesos del servidor.

Gestión de contenedores:

- Cierre los procesos del servidor en respuesta a las indicaciones de Amazon GameLift.
- Informe el estado del contenedor.

Tareas de carga de registros:

- Sube los registros de las sesiones de juego a un bucket de Amazon S3 designado.
- Cargue los registros de los agentes informáticos en un depósito de Amazon S3 designado.

Escalado de la capacidad de la flota

La capacidad de la flota mide el número de sesiones de juego que la flota puede organizar en un momento dado. También puedes medir la capacidad en función del número de jugadores que la flota puede soportar simultáneamente.

Para aumentar o reducir la capacidad de alojamiento de una flota, añades o eliminas instancias de la flota. La estrategia de embalaje de una flota de contenedores determina el número de sesiones de juego que se ejecutan simultáneamente en cada instancia de la flota. Este número indica el número de sesiones de juego (y de espacios para los jugadores) que se suman o restan al aumentar o disminuir la capacidad de la flota.

Con las flotas de contenedores, puedes usar cualquiera de los métodos de escalado proporcionados por Amazon GameLift. Entre ellos se incluyen:

- Establezca la capacidad de la flota manualmente configurando el recuento de instancias de flota específico deseado.
- Configure el escalado automático centrándose en el búfer deseado de instancias disponibles (seguimiento de objetivos). Este método mantiene automáticamente un conjunto de recursos de alojamiento inactivos para que los jugadores entrantes siempre puedan acceder rápidamente a las partidas. A medida que la demanda de los jugadores aumenta o disminuye, el tamaño de este búfer se ajusta continuamente.
- Configura el escalado automático con reglas de escalado personalizadas (función avanzada).

Conexiones cliente/servidor del juego

Las flotas de EC2 gestionadas y las flotas de contenedores gestionan las conexiones entre los clientes de juegos y los servidores de juegos alojados en la nube de forma similar. Cuando Amazon GameLift crea una nueva sesión de juego, el servicio comunica la información de conexión de la sesión de juego. Los clientes del juego utilizan la información para conectarse directamente al servidor del juego que aloja la sesión de juego. Para todos los tipos de flotas, la información de conexión consiste en una dirección IP y una asignación de puertos.

Al crear una flota de contenedores, se definen dos conjuntos de rangos de puertos. En primer lugar, debes definir un rango de puertos de conexión externos que permiten a los clientes del juego conectarse a un juego. En segundo lugar, defines un conjunto de puertos contenedores solo internos, que se asignan a cada proceso del servidor de juegos que se ejecuta en el contenedor. Amazon asigna GameLift dinámicamente los puertos de los contenedores internos a los puertos de conexión externos para que los jugadores puedan acceder a los juegos. Este enfoque proporciona un nivel de seguridad adicional al proteger los servidores de juegos del acceso directo a los puertos de los contenedores.

Al definir los rangos de puertos para una flota de contenedores, debes proporcionar rangos con suficientes puertos para dar cabida a todos los procesos del servidor que se ejecutan simultáneamente en los contenedores de una instancia.

Para tener un control adicional, también debes configurar los permisos de entrada de una flota. Los permisos entrantes determinan qué puertos de conexión están abiertos para el tráfico entrante. Puedes cambiar los permisos de entrada de una flota en cualquier momento. Con los permisos de

entrada, puedes cerrar rápidamente todos los puertos de conexión, abrir algunos o abrir todos, según sea necesario.

Hoja de ruta de desarrollo para contenedores de Amazon GameLift

Esta documentación es para una función que se encuentra en una versión preliminar pública. Está sujeta a cambios.

El siguiente flujo de trabajo resume los pasos para que tus servidores de juegos funcionen en una flota de GameLift contenedores de Amazon.

Paso 1: integra tu juego con Amazon GameLift

Añade funcionalidad a tu servidor de juegos para que pueda comunicarse con el GameLift servicio de Amazon cuando se despliegue en una flota de contenedores. Si utilizas el sistema de FlexMatch emparejamiento, añade esta funcionalidad a tu servidor y cliente del juego. Para obtener información detallada, consulte Integra tu juego con Amazon GameLift.

- Obtén el SDK para GameLift servidores de Amazon (versión 5+) y configúralo con tu proyecto de juego. El SDK del servidor está disponible en C++, C# y Go.
- Modifica el código del servidor de juegos para añadir la funcionalidad requerida del SDK del servidor.
- Package la compilación de su servidor de juegos para Linux. Si estás desarrollando en Windows, es posible que este paso requiera más trabajo para configurar un entorno Linux.
- (Opcional) Prueba la integración de tu servidor de juegos con una GameLift Anywhere flota de Amazon. Pruébelo antes de preparar la imagen de su contenedor para aislar los problemas relacionados con su trabajo de integración. Para probar las conexiones entre el cliente y el servidor del juego, integra también el cliente del juego.

> **Note**
>
> Si estás desarrollando en Windows, configura un espacio de trabajo Linux independiente o usa una herramienta como el subsistema de Windows para Linux (WSL). Necesitarás un entorno Linux para probar la versión de tu servidor de juegos y también para crear y probar las imágenes de tu contenedor.

Paso 2: Prepara la imagen del contenedor del servidor de juegos

Crea una imagen contenedora que ejecute los procesos de tu servidor de juegos y guárdala en un repositorio de Amazon Elastic Container Registry (Amazon ECR) para usarla con Amazon. GameLift Para obtener instrucciones detalladas, consulte Prepara una imagen de contenedor con el software de tu servidor de juegos.

- Configura un directorio de trabajo para la imagen del contenedor, con la compilación del juego de Linux, el script de instalación y todo el software y las dependencias compatibles.
- Obtén el código fuente de Amazon GameLift Agent, compílalo y añade el `jar` archivo a tu directorio de trabajo.
- Obtén el Dockerfile predeterminado y modifícalo para configurar una imagen de contenedor con el software de tu servidor de juegos.
- Crea tu imagen de contenedor. Realice este paso en un entorno Linux.
- Cree un repositorio privado de Amazon ECR e inserte en él la imagen de su contenedor. Cree el repositorio en el mismo Cuenta de AWS Región de AWS lugar donde planea desplegar su flota de contenedores.
- (opcional) Pruebe las imágenes de sus contenedores con su Anywhere flota. Puedes establecer una configuración de tiempo de ejecución para pasar instrucciones al Amazon GameLift Agent.

Paso 3: Crea tus contenedores y grupos de contenedores

Diseña una arquitectura de contenedores para el alojamiento de juegos en Amazon GameLift. Consulte Diseña una flota de GameLift contenedores de Amazon y Crea definiciones de grupos de contenedores para una flota de GameLift contenedores de Amazon.

- Defina las configuraciones de sus contenedores. Para cada contenedor, definirá aspectos como los procesos de tiempo de ejecución, la asignación de memoria, las comprobaciones de estado, los puertos de red, etc.
- Usa la GameLift consola de Amazon o la AWS CLI para crear definiciones de grupos de contenedores con tus configuraciones de contenedores. Al crear una definición de grupo de contenedores, Amazon GameLift toma una instantánea de cada imagen de contenedor en ese momento.

Paso 4: Despliega tu servidor de juegos en contenedores en una flota de contenedores

Usa las definiciones de grupos de contenedores creadas en el paso anterior para crear una flota de contenedores e implementar tu software de servidor de juegos en contenedores. Consulte Crea una flota de GameLift contenedores de Amazon.

- Usa la GameLift consola de Amazon o la AWS CLI para crear una flota de contenedores.
- Realice un seguimiento del estado de la flota a medida que se despliegan y activan las instancias de la flota Compruebe los eventos de creación de la flota para comprobar que la flota se está desplegando correctamente en todas las ubicaciones.
- Comprueba que los clientes del juego puedan solicitar sesiones de juego y unirse a ellas y jugarlas. Si has configurado el matchmaking, prueba esos escenarios.

Paso 5: Gestiona tus flotas

Mientras se prepara para el uso a nivel de producción, diseñe su solución de alojamiento de juegos y gestione su ciclo de vida de alojamiento.

- Crea flotas en varias ubicaciones y flotas en otras Regiones de AWS para apoyar a tu base de jugadores.
- Configura la ubicación del alojamiento de juegos con colas o emparejamientos. FlexMatch Consulta estos recursos:
  - Configuración de colas de Amazon GameLift para la ubicación de las sesiones de juego
  - FlexMatch Developer Guide
- Configura el escalado automático para gestionar la capacidad de la flota en función de la demanda de los jugadores en las sesiones de juego.
- Configure el monitoreo de sus flotas de contenedores. Trabaje con GameLift las métricas de Amazon, recupere los registros de las sesiones de juego y los registros de contenedores, configure el acceso remoto a contenedores individuales.
- Configure una gestión a largo plazo de las flotas de contenedores. Utilice los alias de flota para agilizar el proceso de actualización de las flotas de contenedores. Cree AWS CloudFormation plantillas para gestionar el ciclo de vida de la flota. Consulta estos recursos:
  - Añadir un alias a una GameLift flota de Amazon
  - Administración de recursos mediante AWS CloudFormation

Integra tu juego con Amazon GameLift

Esta documentación corresponde a una función que se encuentra en una versión preliminar pública. Está sujeta a cambios.

Antes de que puedas crear una imagen de contenedor con el software de tu servidor de juegos e implementarla en Amazon GameLift como alojamiento en la nube, primero integra tu proyecto

de juego con el SDK GameLift del servidor de Amazon y crea un servidor de juegos para que se ejecute en Linux. En este tema se presentan las distintas herramientas de integración que GameLift proporciona Amazon.

Los servidores de juegos alojados deben poder comunicarse con el GameLift servicio de Amazon. Configura la comunicación añadiendo el SDK GameLift del servidor de Amazon (versión 5+) a tu proyecto de juego y modificando el código del servidor del juego. Amazon GameLift proporciona recursos y documentación del SDK del servidor para admitir varios idiomas y motores de juego.

El proceso de integración de los servidores de juegos en contenedores es prácticamente idéntico al de la integración de servidores de juegos para el alojamiento en flotas gestionadas de EC2 o Amazon. GameLift Anywhere

Herramientas de integración

Amazon GameLift proporciona las siguientes herramientas y soporte de idiomas para la integración:

Para desarrolladores de Unreal Engine

Usa el complemento ligero para Unreal. Este complemento incluye las bibliotecas SDK del servidor C++ con la GameLift funcionalidad requerida de Amazon. Usa la documentación para configurar tu proyecto de juego de Unreal para el complemento y actualiza el código del juego con los bloques de código proporcionados para añadir las funciones necesarias a las versiones de servidor y cliente.

- Descarga el plugin del SDK
- Guía: integra tu proyecto de Unreal con Amazon GameLift
- Guía de referencia: C++ Server SDK 5 para Unreal

Nota: El complemento GameLift independiente de Amazon para Unreal Engine no admite el uso de flotas de contenedores.

Para desarrolladores de Unity

Usa el complemento ligero para Unity. Este complemento incluye las bibliotecas del SDK del servidor C# con la GameLift funcionalidad requerida de Amazon. Usa la documentación para configurar tu proyecto de juego de Unreal para el complemento y actualiza el código del juego con los bloques de código proporcionados para añadir las funciones necesarias a las versiones de servidor y cliente.

- Descarga el plugin del SDK

- Guía: integra tu proyecto de Unity con Amazon GameLift
- Guía de referencia: C# Server SDK 5 para Unity

Nota: El complemento GameLift independiente de Amazon para Unity no admite el uso de flotas de contenedores.

Para desarrolladores que utilizan otros motores de juegos

Siga esta guía general de integración de servidores y clientes:

- Integre un servidor de juegos
- Integre un cliente de juego

Amazon GameLift ofrece bibliotecas del SDK 5 de servidor para los siguientes idiomas:

- Server SDK 5 para C++ [descarga del SDK] [Guía de referencia]
- Server SDK 5 para C# [descarga del SDK] [Guía de referencia]
- Server SDK 5 for Go [descarga del SDK] [Guía de referencia]

Construye tu servidor de juegos para Linux

Las flotas de GameLift contenedores de Amazon admiten servidores de juegos que se ejecutan en una plataforma Linux. Estos son algunos consejos para crear un servidor de juegos para un destino Linux:

- Si estás desarrollando tu juego con el motor de juegos Unity, el editor de juegos incluye soporte integrado sin requisitos especiales de compilación para Linux.
- Si estás desarrollando tu juego en C++, debes incluir las bibliotecas OpenSSL para Linux cuando crees el GameLift Amazon Server SDK para C++ y cuando construyas tu servidor de juegos. Incluye también las mismas bibliotecas en la imagen del contenedor del servidor de juegos.
- Si estás desarrollando tu juego con Unreal Engine en Windows, ten en cuenta estas opciones:
  - Trabaja con Unreal Engine para configurar una cadena de herramientas de compilación cruzada.
  - Configure un espacio de trabajo de Linux independiente o utilice una herramienta como el subsistema de Windows para Linux (WSL). Puedes usar este entorno para ejecutar el Unreal Editor en Linux y crear tu servidor de juegos.

Pon a prueba tu integración localmente

Puedes probar la integración de tu juego de forma local con una GameLift Anywhere flota de Amazon. Este enfoque es una buena práctica para ayudar a aislar los problemas directamente relacionados con la integración. Una Anywhere flota es una herramienta útil para ejecutar aplicaciones de prueba y escenarios de juego, como iniciar o detener sesiones de juego y rastrear las conexiones de los jugadores. Puedes construir y probar de forma iterativa mucho más rápido con una Anywhere flota, lo que ofrece una mayor visibilidad de la actividad de los anfitriones.

Consulta Realización de una prueba de integración con flotas de Amazon GameLift Anywhere para obtener ayuda sobre el uso de una GameLift Anywhere flota de Amazon para las pruebas de integración. El flujo de trabajo para configurar un entorno de pruebas es el siguiente:

1. Configura un dispositivo local que ejecute Linux.
2. Configure una Anywhere flota. Cree una ubicación personalizada para su dispositivo local, cree una Anywhere flota y, a continuación, registre su dispositivo local como cómputo de la flota.
3. Obtén un token de autenticación para tu servidor de juegos. El proceso de servidor integrado requiere un token para autenticarse con el GameLift servicio de Amazon. Puede reutilizar el mismo token para varios procesos de servidor que se ejecuten simultáneamente. Este paso solo es necesario cuando se utiliza una Anywhere flota para las pruebas de integración.

   > **Note**
   >
   > Los tokens de autenticación son temporales y deben actualizarse periódicamente. Considere la posibilidad de añadir un script al paquete de compilación del servidor para solicitar un nuevo token.

4. Actualiza el código de tu servidor de juegos paraAnywhere. Cuando se ejecuta en una flota de Anywhere, el servidor del juego debe llamar a la acción del SDK del servidor `InitSdk()` (C++) (C#) (Unreal) con los siguientes parámetros del servidor. Este paso solo es necesario cuando se utiliza una Anywhere flota para realizar pruebas de integración. Una vez que añadas el Amazon GameLift Agent a la imagen de tu contenedor, gestionará estos parámetros automáticamente.

   Como práctica recomendada, configure el código del servidor para que extraiga estos valores de las variables de entorno o de los argumentos de la consola que especifique en el lanzamiento.

   - `webSocketUrl`— Usa el valor de`GameLiftServiceSdkEndpoint`, que se devuelve de la llamada `aregister-compute`.

- `processId`— Asigne un identificador único para el proceso del servidor.
- `fleetId`— El identificador de flota de Anywhere, que se devuelve de la llamada `acreate-fleet`.
- `authToken`— Un token de autenticación válido, que se devuelve de la llamada `aget-compute-auth-token`.

5. En tu máquina local, configura el software de compilación del servidor de juegos e inicia un proceso de servidor.

   Si la integración del servidor se realiza correctamente, el proceso del servidor llama `InitSDK()` a la acción del SDK del servidor para establecer la conexión con el GameLift servicio de Amazon, seguida de una llamada `ProcessReady()` para notificar al servicio que está preparado para albergar una sesión de juego.

6. Inicia una sesión de juego. Si has integrado tu cliente de juego para solicitar una sesión de juego, puedes usarlo para solicitar una nueva sesión de juego. Si no es así, utilice el comando AWS CLI create-game-session. Amazon GameLift crea un `GameSession` objeto e inicia el proceso para iniciar una nueva sesión de juego.

   Si tu integración funciona, Amazon GameLift llama a un proceso de servidor de tu estación de trabajo local para iniciar una nueva sesión de juego (mediante la función `onStartGameSession()` callback). Cuando una sesión de juego está lista para los jugadores, el servidor procesa las llamadas. `ActivateGameSession()` En respuesta, Amazon GameLift actualiza el `GameSession` estado y la información de conexión para que el cliente del juego pueda conectarse a la sesión de juego y jugarlo.

Prepara una imagen de contenedor con el software de tu servidor de juegos

Esta documentación es para una función que se encuentra en una versión preliminar pública. Está sujeta a cambios.

El contenedor es el elemento más básico de la flota de GameLift contenedores de Amazon. Tu contenedor incluye tu servidor de juegos, junto con sus dependencias, como los SDK, el software, los directorios y los archivos.

Para funcionar en una flota de contenedores, tu servidor de juegos debe funcionar en Linux y estar integrado con el SDK 5.x del servidor.

Temas

Configura tu directorio de trabajo

Tu directorio de trabajo es donde colocas todos los archivos que necesitas para crear la imagen de tu contenedor y definir cómo Amazon la GameLift ejecuta.

Para configurar el directorio de trabajo de tu contenedor

1. Crea el directorio en el que quieres trabajar con las imágenes de tus GameLift contenedores de Amazon.

   Example

   Por ejemplo:

   ```
   [~/]$ mkdir -p work/glc/gamebuild && cd work && find .
   .
   ./glc
   ./glc/gamebuild
   ```

2. Clona el Amazon GameLift Agent.

   Example

   Por ejemplo:

   ```
   [~/work]$ git clone https://github.com/aws/amazon-gamelift-agent.git
   Cloning into 'amazon-gamelift-agent'...
   ```

3. Construye el GameLiftAgent con Maven.

   Example

   Por ejemplo:

   ```
   [~/work]$ cd amazon-gamelift-agent
   ```

Example

```
[~/work/amazon-gamelift-agent]$ mvn clean compile assembly:single && \
mv target ../glc && cd .. && find glc
```

4. Agregue un servidor de juegos que se haya integrado con el SDK 5.x del servidor, creado y empaquetado en un `.ZIP` archivo.
5. Copia tu `.ZIP` archivo a. `~/work/glc/gamebuild/`

Si no tienes un servidor de juegos con el SDK 5.x, puedes descargar y usar nuestro SimpleServerjuego de muestra para probar a usar una flota de contenedores.

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
```

Crea tu imagen de contenedor

El Dockerfile especifica el entorno, el software y las instrucciones para crear el contenedor.

Para crear su Dockerfile

1. Ve al subdirectorio. `glc`

   Example

   ```
   [~/work]$ cd glc && find
   ```

```
.
./target
./target/GameLiftAgent-1.0.jar
./gamebuild
```

2. Cree y abra un Dockerfile nuevo.

   Example

   Por ejemplo:

   ```
   [~/work/glc]$ nano Dockerfile
   ```

3. Copia una de las siguientes plantillas y, a continuación, pega el contenido en tu Dockerfile.

Plantilla de Dockerfile para tu servidor de juegos

Esta plantilla contiene las instrucciones mínimas que un contenedor necesita para poder utilizarse en una GameLift flota de Amazon. Modifica el contenido según sea necesario para tu servidor de juegos.

```
# 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 para la muestra **`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**
>
> Nota: Algunas de las variables de entorno del Dockerfile pueden ser anuladas por. ContainerDefinition

Para crear la imagen de tu contenedor

1. Cree la imagen de su contenedor.

   Si utilizas tu propio servidor SDK 5.x

   Puedes especificar el nombre del repositorio local que desees.

   Example

   ```
   [~/work/glc]$ docker build -t <local repository name>:<optional tag> .
   ```

   Si estás usando nuestro **`SimpleServer`** ejemplo

   Example

   ```
   [~/work/glc]$ docker build -t simple-server:version-1 .
   Successfully built 0123456789012
   Successfully tagged simple-server:version-1
   ```

   > **Note**
   >
   > En los siguientes ejemplos, utilizamos *`simpler-server`* como `REPOSITORY` valor inicial y `version-1` como valor. `TAG`

2. Vea la lista de imágenes y anote los valores `REPOSITORY` y`IMAGE ID`. Los necesitará en un procedimiento que se indica a continuación.

   Example

```
[~/work/glc]$ docker images
REPOSITORY       TAG          IMAGE ID        CREATED           SIZE
simple-server    version-1    0123456789012    14 minutes ago    1.24GB
```

Envía la imagen de tu contenedor a Amazon ECR

Cargue la imagen del contenedor en un repositorio privado de Amazon ECR. Cuando creas una definición de grupo de contenedores, haces referencia a esta ubicación del repositorio para que Amazon GameLift pueda tomar una instantánea de la imagen de tu contenedor y usarla al desplegar una flota de contenedores.

> **Note**
>
> Si aún no tiene un repositorio privado de Amazon ECR, cree uno.

Para obtener sus credenciales de Amazon ECR

- Antes de poder enviar la imagen de su contenedor a Amazon ECR, debe adquirir sus AWS credenciales de forma temporal y proporcionárselas a Docker. Obtenga sus credenciales de Amazon ECR para que Docker pueda iniciar sesión.

  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
```

Para enviar la imagen del contenedor a Amazon ECR

1. Copie el URI del repositorio privado de Amazon ECR que desee usar.
2. Aplica una etiqueta Amazon ECR a la imagen de tu contenedor.

   Example

   ```
   [~/work/glc]$ docker tag <IMAGE ID from above> <Amazon ECR private repository
    URI>:<optional tag>
   ```

3. Envía la imagen de tu contenedor a Amazon ECR

   Example

   ```
   [~/work/glc]$ docker image push  <Amazon ECR private repository URI>
   ```

Diseña una flota de GameLift contenedores de Amazon

Esta documentación corresponde a una función que se encuentra en una versión preliminar pública. Está sujeta a cambios.

En estos temas se presentan las decisiones clave que tomarás al configurar una flota de GameLift contenedores de Amazon. Tus decisiones afectan a la forma en que configuras los ajustes de los contenedores, los grupos de contenedores y las flotas.

Temas

- Diseñe la estructura de contenedores de su flota
- Establece límites de recursos
- Designe los contenedores esenciales
- Configure las conexiones de red
- Configure los controles de estado de los contenedores
- Establezca las dependencias de los contenedores
- Configure una flota de contenedores

Diseñe la estructura de contenedores de su flota

Como primer paso, identifica el software y los recursos necesarios para alojar tu servidor de juegos, incluidos los siguientes:

- Tu aplicación de servidor de juegos. La aplicación debe estar integrada con las GameLift funciones de alojamiento de Amazon, incluida la versión 5+ del SDK del servidor. Consulte Integra tu juego con Amazon GameLift.
- El GameLift agente de Amazon. Este agente informático mantiene la comunicación con el GameLift servicio de Amazon y gestiona el ciclo de vida de todos los procesos del servidor de juegos. Para obtener más información, consulte Servidores de juegos y Amazon GameLift Agent.
- Software y recursos adicionales según sea necesario. Esto podría incluir el software necesario para ejecutar las aplicaciones del servidor de juegos. El software de apoyo más común se utiliza para el registro y la supervisión, la seguridad, la entrega de contenido y la sincronización de datos.

A continuación, decide cómo estructurar el software y los recursos para una flota de GameLift contenedores de Amazon. Amazon GameLift utiliza grupos de contenedores para organizar los contenedores. Una flota siempre tiene un grupo de réplicas de contenedores y, si lo desea, puede tener una flota de contenedores demoníaca. Para obtener más información, consulte Componentes de la flota de contenedores.

- Comience por diseñar su grupo de contenedores de réplicas. Tenga en cuenta estas directrices:
  - Agrupa tu aplicación de servidor de juegos y el Amazon GameLift Agent en el mismo contenedor. Convierte este contenedor en el único contenedor esencial del grupo de réplicas.
  - Organice el resto del software de su servidor de juegos en contenedores. Puedes optar por poner todo en un único contenedor en el grupo de réplicas. O puede optar por crear uno o más contenedores con sidecar. Algunas de las razones para usar un sidecar son las siguientes:
    - Para configurar una secuencia de inicio/apagado para un software individual. Puede lograrlo colocando el software en contenedores separados y configurando las dependencias entre ellos.
    - Para establecer límites específicos de los contenedores para el uso de memoria y CPU.
    - Para especificar diferentes ajustes de configuración de contenedores para cada contenedor, como un comando de lanzamiento, un punto de entrada, un directorio de trabajo, variables de entorno o comprobaciones de estado.
- Decida si necesita un grupo daemon de contenedores para su flota. Considere lo siguiente:

- Los contenedores Daemon se utilizan normalmente para ejecutar procesos en segundo plano o de supervisión.
- Los contenedores de un grupo de demonios no se replican en una instancia de flota. Esto significa que los contenedores de un grupo de demonios no escalan junto con el grupo de contenedores de réplicas.
- Un grupo de demonios puede tener varios contenedores. Puede designar cualquier contenedor de un grupo de demonios como esencial.

Establece límites de recursos

Para cada grupo de contenedores, determine la cantidad de memoria y CPU que necesita el grupo para ejecutar su software. Amazon GameLift se basa en esta información para gestionar los recursos del grupo de contenedores. También utiliza esta información para calcular cuántos grupos de réplicas de contenedores puede contener una imagen de flota. También puede establecer límites para contenedores individuales.

Establece límites opcionales para los contenedores

Establecer límites de recursos específicos para cada contenedor te permite ejercer un mayor control sobre cómo los contenedores individuales pueden usar los recursos del grupo. Si no estableces límites específicos para los contenedores, todos los contenedores del grupo comparten los recursos del grupo. El uso compartido ofrece una mayor flexibilidad para usar los recursos donde se necesiten. También aumenta la posibilidad de que los procesos compitan entre sí y provoquen la falla del contenedor.

Defina cualquiera de las siguientes `ContainerDefinition` propiedades para cualquier contenedor.

- `SoftLimit`(memoria): reserve una cantidad mínima de memoria para el uso exclusivo del contenedor. El contenedor siempre tiene disponible la cantidad reservada. Puede superar este mínimo en cualquier momento, si hay recursos adicionales disponibles.
- `HardLimit`(memoria): establezca un límite máximo de memoria para el contenedor. Si el contenedor supera este límite, se reinicia.
- `Cpu`límite: reserve una cantidad mínima de recursos de CPU para el uso exclusivo del contenedor. El contenedor siempre tiene disponible la cantidad reservada. Puede superar este mínimo en cualquier momento, si hay recursos adicionales disponibles. (1024 unidades de CPU equivalen a 1 vCPU).

Establezca los límites totales de recursos para un grupo de contenedores

Indique a Amazon GameLift cuántos recursos de memoria y CPU necesita cada grupo de contenedores. El objetivo es asignar recursos suficientes para optimizar el rendimiento del servidor de juegos. Amazon GameLift utiliza estos límites para calcular cómo empaquetar grupos de réplicas de contenedores en una instancia de flota. También los usarás al elegir un tipo de instancia para una flota de contenedores.

Calcula la memoria y la CPU totales necesarias para todos los procesos de cada contenedor de un grupo. Considere lo siguiente:

- ¿Qué procesos se ejecutan en todos los contenedores del grupo de contenedores? Sume los recursos necesarios para estos procesos.
- ¿Cuántos procesos simultáneos del servidor de juegos piensas ejecutar en cada grupo de contenedores? Establece este valor como parte de la configuración del tiempo de ejecución de una flota, pero aquí debe planificar suficiente memoria para ellos (consulte Optimizar la configuración del tiempo de ejecución).

En función de su estimación de los requisitos del grupo de contenedores, defina las siguientes `ContainerGroupDefinition` propiedades:

- `TotalMemoryLimit`— Establezca un límite máximo de memoria para el grupo de contenedores. Todos los contenedores del grupo comparten la memoria asignada. Si estableces límites de contenedores individuales, el límite total de memoria debe ser:
  - igual o superior a la suma de todos los límites de memoria flexible de los contenedores
  - igual o superior al límite máximo de memoria dura de un contenedor del grupo
- `TotalCpuLimit` — Establezca un límite máximo de CPU para el grupo de contenedores. Todos los contenedores del grupo comparten los recursos de CPU asignados. Si estableces límites de contenedores individuales, el límite total de CPU debe ser:
  - igual o superior a la suma de todos los límites de CPU de los contenedores. Como práctica recomendada, considere configurar este valor para que duplique la suma de los límites de CPU del contenedor.

Escenario de ejemplo

Supongamos que estamos definiendo un grupo de contenedores de réplicas con los tres contenedores siguientes:

- El contenedor A es nuestro contenedor de réplica esencial. Ejecuta los procesos del servidor del juego y el Amazon GameLift Agent. Estimamos los requisitos de recursos para un servidor

de juegos en 512 MiB y 1024 CPU. Planeamos que el contenedor ejecute 10 procesos de servidor. Como en este contenedor se ejecuta nuestro software más crítico, hemos establecido una reserva de memoria flexible de 6144 MiB sin límite de memoria dura ni límite de reserva de CPU.

- El contenedor B ejecuta software de soporte con requisitos de recursos estimados en 1024 MiB y 1536 CPU. Hemos establecido un límite de reserva de memoria flexible de 1024 MiB, un límite de memoria dura de 2048 MiB y un límite de reserva de CPU de 1024 CPU.
- El contenedor C ejecuta utilidades de registro no críticas y otras utilidades de monitoreo. Hemos establecido un límite de memoria dura de 512 MiB y un límite de reserva de CPU de 512 CPU.

Con esta información, establecemos los siguientes límites totales para el grupo de contenedores:

- Límite total de memoria: 7680 MiB. Este valor supera (1) la suma de los límites de memoria blanda (6144+1024 MiB) y (2) el límite máximo de memoria dura (1024 MiB).
- Límite total de CPU: 13312 CPU. Este valor supera la suma del límite de CPU (1024+512 CPU).

Designe los contenedores esenciales

Para cada recipiente, dedíquelo como esencial o no esencial. Todos los grupos de contenedores deben tener al menos un contenedor esencial. El contenedor esencial realiza el trabajo fundamental del grupo de contenedores, como alojar los servidores de juegos. Se espera que el contenedor esencial esté siempre en funcionamiento. Si se produce un error, se reinicia todo el grupo de contenedores.

- El grupo de réplicas de contenedores de su flota puede tener exactamente un contenedor esencial. Este contenedor ejecuta el Amazon GameLift Agent y los procesos que gestiona el servidor del juego.
- Si tu flota tiene un grupo de contenedores demoníaco, puedes designar varios contenedores esenciales. Haga que un contenedor daemon sea imprescindible si quiere que un fallo en un contenedor provoque el reinicio del grupo de contenedores.

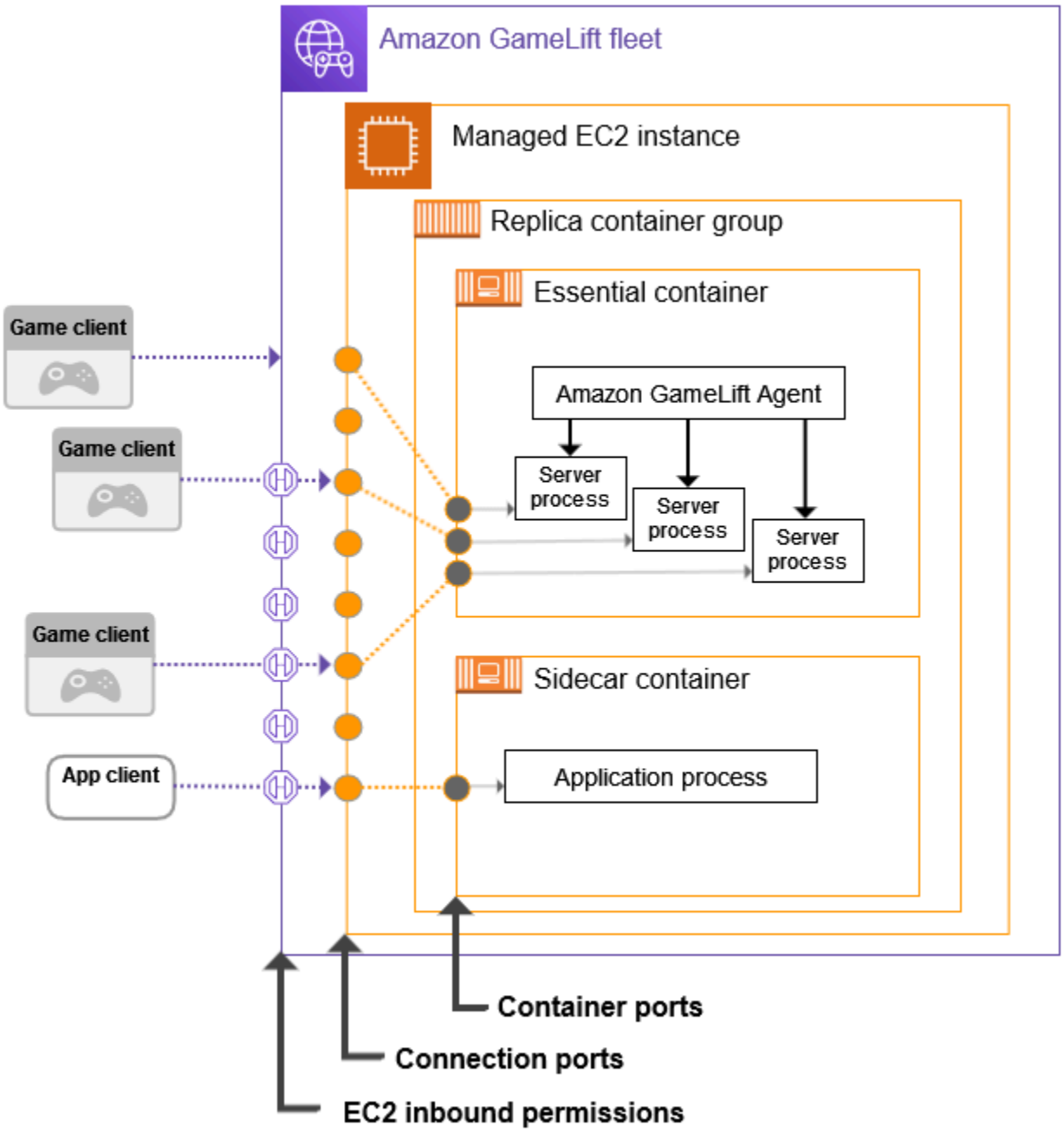
Defina la `ContainerDefinition` propiedad `Essential` en verdadero o falso para cada contenedor.

Configure las conexiones de red

Puede establecer el acceso a la red para permitir que el tráfico externo se conecte a cualquier contenedor de una flota de contenedores. Por ejemplo, debes establecer conexiones de red con el contenedor que ejecuta los procesos del servidor de juegos para que los clientes del juego puedan unirse y jugar a tu juego. Los clientes de juegos se conectan a los servidores de juegos mediante puertos y direcciones IP.

En una flota de contenedores, la conexión entre un cliente y un servidor no es directa. Internamente, un proceso de un contenedor escucha en un puerto de contenedores. Externamente, el tráfico entrante se conecta a una instancia de flota mediante un puerto de conexión. Amazon GameLift mantiene las asignaciones entre los puertos de contenedores internos y los puertos de conexión externos para que el tráfico entrante se dirija al proceso correcto de la instancia.

Amazon GameLift proporciona un nivel adicional de control para tus conexiones de red. Cada flota de contenedores tiene una configuración de permisos de entrada, que te permite controlar el acceso a cada puerto de conexión externo. No puedes cambiar la configuración de los puertos de una flota existente, pero puedes permitir o restringir el acceso según sea necesario ajustando los permisos de entrada. Por ejemplo, puedes eliminar los permisos de todos los puertos de conexión para impedir el acceso a los contenedores de la flota.



Establezca los rangos de puertos de los contenedores

Configure una definición de contenedor con suficientes puertos de contenedor para cualquier proceso que necesite acceso externo. Algunos contenedores no necesitarán ningún puerto. Otros deben tener puertos suficientes para asignar uno a cada proceso que lo necesite.

El grupo de contenedores de réplicas esencial, en el que se ejecutan los servidores de juegos, necesita un puerto para cada proceso de servidor de juegos que se ejecute simultáneamente (tal y como está configurado en la flota`RuntimeConfiguration`). El proceso del servidor del juego escucha en el puerto asignado y lo informa a Amazon GameLift.

Cuando crees una definición de grupo de contenedores, define un rango de puertos para cada contenedor que necesite acceso a la red (consulta ContainerDefinitionInput: PortConfiguration). Asegúrese de que el rango sea lo suficientemente grande como para asignar un puerto a cada proceso que lo necesite. Los procesos deben tener asignados números de puerto en la configuración de puertos del contenedor.

Establezca los rangos de puertos de conexión

Configure su flota de contenedores con un conjunto de puertos de conexión. Los puertos de conexión proporcionan acceso externo a las instancias de la flota en las que se utilizan sus contenedores. Amazon GameLift asigna los puertos de conexión y los asigna a los puertos de contenedores según sea necesario.

Al crear una flota de contenedores, defina un rango de puertos de conexión (consulte ContainerGroupsConfiguration: ConnectionPortRange). Asegúrese de que el rango tenga suficientes puertos para asignarlos a todos los puertos de contenedores de una instancia de flota. Para calcular los puertos de conexión mínimos necesarios, usa la siguiente fórmula:

```
[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]
```

Como práctica recomendada, duplique el número mínimo de puertos de conexión.

> **Note**
>
> La cantidad de puertos de conexión puede limitar potencialmente la cantidad de grupos de contenedores de réplicas por instancia. Si una flota solo tiene puertos de conexión suficientes para un grupo de contenedores de réplicas por instancia, Amazon GameLift implementará solo un grupo de contenedores de réplicas, incluso si las instancias tienen suficiente potencia de cómputo para varios grupos de contenedores de réplicas.

Configure los permisos de entrada

Los permisos de entrada controlan el acceso externo a una flota de contenedores especificando qué puertos de conexión se deben abrir para el tráfico entrante. Puedes usar esta configuración para activar y desactivar el acceso a la red de una flota según sea necesario.

Al crear una flota de contenedores, defina un conjunto de permisos de entrada (consulte: CreateFleet EC2). InboundPermissions Defina las propiedades de los puertos de permisos de entrada para incluir algunos o todos los valores de la configuración del puerto de conexión de la flota. Para cambiar los permisos de entrada de una flota de contenedores existente, llama. UpdateFleetPortSettings

Escenario de ejemplo

Este ejemplo ilustra cómo configurar las tres propiedades de conexión de red.

- El grupo de réplicas de contenedores de nuestra flota tiene 1 contenedor, que ejecuta los procesos del servidor del juego. La configuración del tiempo de ejecución indica al contenedor que ejecute 10 procesos simultáneos del servidor del juego.

  En la definición del grupo de contenedores de réplicas, establecemos el `PortConfiguration` parámetro para este contenedor de la siguiente manera:

  ```
  "PortConfiguration": {
    "ContainerPortRanges": [ { "FromPort": 10, "ToPort": 20, "Protocol": "TCP"} ]
    }
  ```

- Nuestra flota también tiene un grupo de contenedores daemon con 1 contenedor. Tiene 1 proceso que necesita acceso a la red. En la definición del grupo de contenedores daemon, configuramos el `PortConfiguration` parámetro para este contenedor de la siguiente manera:

  ```
  "PortConfiguration": {
    "ContainerPortRanges": [ { "FromPort": 25, "ToPort": 25, "Protocol": "TCP"} ]  }
  ```

- Nuestra flota está configurada con 3 grupos de réplicas de contenedores por instancia de flota. Con esta información, podemos usar la fórmula para calcular el número de puertos de conexión que necesitamos:
  - Mínimo: 31 puertos [10 puertos de contenedores de réplicas * 3 grupos de contenedores de réplicas por instancia + 1 puerto de contenedor daemon]
  - Práctica recomendada: 62 puertos [puertos mínimo* 2]

Al crear la flota de contenedores, configuramos el `ConnectionPortRange` parámetro de la `ContainerGroupsConfiguration` siguiente manera:

```
"ConnectionPortRange": { "FromPort": 1010, "ToPort": 1071 }
```

- Queremos permitir el acceso a todos los puertos de conexión disponibles. Al crear la flota de contenedores, configuramos el `EC2InboundPermissions` parámetro de la siguiente manera:

```
"EC2InboundPermissions": [
  {"FromPort": 1010, "ToPort": 1071, "IpRange": "10.24.34.0/23", "Protocol":
 "TCP"} ]
```

Configure los controles de estado de los contenedores

Un contenedor se reinicia automáticamente si sufre una falla en la terminal y deja de funcionar. Si el contenedor es esencial, se reinicia todo el grupo de contenedores.

Puede definir criterios personalizados adicionales para medir el estado del contenedor y utilizar una comprobación de estado para probar esos criterios. Para configurar una comprobación del estado de un contenedor, puede definirla en una imagen de contenedor de docker o en su definición de contenedor. Si configuras una comprobación de estado en la definición del contenedor, anulará cualquier configuración de la imagen del contenedor.

Defina las comprobaciones de estado opcionales en función del tipo de contenedor de la siguiente manera:

- En el caso de un contenedor de réplica esencial, no configure las comprobaciones de estado. El GameLift agente de Amazon gestiona automáticamente los informes de estado de este contenedor.
- En el caso de los contenedores de réplica no esenciales y de cualquier contenedor daemon, si lo desea, puede configurar los parámetros de comprobación de estado.

Defina las siguientes `ContainerDefinition` propiedades para la comprobación del estado de un contenedor:

- `Command`— Proporcione un comando que compruebe algún aspecto del estado del contenedor. Usted decide qué criterios utilizar para medir la salud. El comando debe dar como resultado un valor de salida de 1 (en mal estado) o 0 (en buen estado).
- `StartPeriod`— Especifique un retraso inicial antes de que los fallos en las comprobaciones de estado empiecen a contabilizarse. Este retraso da tiempo al contenedor para iniciar sus procesos.
- `Interval`— Decida con qué frecuencia se va a ejecutar el comando de comprobación de estado. ¿Con qué rapidez desea detectar y resolver un fallo en un contenedor?
- `Timeout`— Decida cuánto tiempo esperar para que se ejecute correctamente o no antes de volver a intentar ejecutar el comando de comprobación de estado. ¿Cuánto tiempo debe tardar en completarse el comando de comprobación de estado?
- `Retries`— ¿Cuántas veces se debe volver a intentar el comando de verificación de estado antes de que se registre un error?

Establezca las dependencias de los contenedores

Dentro de cada grupo de contenedores, puede establecer las dependencias entre los contenedores en función del estado del contenedor. Una dependencia afecta al momento en que el contenedor dependiente puede iniciarse o cerrarse en función del estado de otro contenedor.

Un caso de uso clave de las dependencias es crear secuencias de inicio y cierre para el grupo de contenedores.

Por ejemplo, es posible que desee que el contenedor A comience primero y se complete correctamente antes de que se inicien los contenedores B y C. Para lograrlo, cree primero una dependencia para el contenedor B en el contenedor A, con la condición de que el contenedor A se complete correctamente. A continuación, cree una dependencia para el contenedor C en el contenedor A con la misma condición. Las secuencias de inicio se producen en orden inverso al de cierre.

Configure una flota de contenedores

Al crear una flota de contenedores, tenga en cuenta los siguientes puntos de decisión. La mayoría de estos puntos dependen de la arquitectura y configuración del contenedor.

Decida dónde quiere desplegar su flota

> En general, querrás desplegar tus flotas geográficamente cerca de tus jugadores para minimizar la latencia. Puedes desplegar tu flota de contenedores en cualquier Each Región de AWS

GameLift compatible con Amazon. Si quieres implementar el mismo servidor de juegos en ubicaciones geográficas adicionales, puedes añadir ubicaciones remotas a la flota, incluidas Regiones de AWS las Zonas Locales. En el caso de una flota con varias ubicaciones, puedes ajustar la capacidad de forma independiente en cada ubicación de la flota. Para obtener más información sobre las ubicaciones de flota compatibles, consulteUbicaciones GameLift de alojamiento de Amazon.

Elija un tipo y un tamaño de instancia para su flota

Amazon GameLift admite una amplia gama de tipos de instancias de Amazon EC2, todas las cuales están disponibles para su uso con una flota de contenedores. La disponibilidad y el precio del tipo de instancia varían según la ubicación. Puedes ver una lista de los tipos de instancias compatibles, filtrados por ubicación, en la GameLift consola de Amazon (en Recursos, instancias y cuotas de servicio).

Al elegir un tipo de instancia, ten en cuenta primero la familia de instancias. Las familias de instancias ofrecen varias combinaciones de capacidades de CPU, memoria, almacenamiento y red. Obtenga más información sobre las familias de instancias EC2. Dentro de cada familia, puede elegir entre una variedad de tamaños de instancia. Tenga en cuenta los siguientes aspectos al seleccionar un tamaño de instancia:

- ¿Cuál es el tamaño mínimo de instancia que puede soportar tu carga de trabajo? Usa esta información para eliminar cualquier tipo de instancia que sea demasiado pequeña.
- ¿Qué tamaños de tipos de instancias son adecuados para su arquitectura de contenedores? Lo ideal es elegir un tamaño que pueda alojar varias copias de su grupo de contenedores de réplicas con un mínimo de espacio desperdiciado.
- ¿Qué granularidad de escalado tiene sentido para tu juego? Ampliar la capacidad de la flota implica añadir o eliminar instancias, y cada instancia representa la capacidad de albergar un número específico de sesiones de juego. Ten en cuenta la cantidad de capacidad que deseas añadir o eliminar con cada instancia. Si la demanda de los jugadores varía en miles de un minuto a otro, podría ser conveniente utilizar instancias muy grandes que puedan alojar cientos o miles de sesiones de juego. Por el contrario, es posible que prefieras un control de escalado más detallado con tipos de instancias más pequeñas.
- ¿Hay ahorros de costos disponibles en función del tamaño? Es posible que el costo de algunos tipos de instancias varíe según la ubicación debido a la disponibilidad.

Optimice la configuración del tiempo de ejecución

La configuración del tiempo de ejecución de una flota es un conjunto de instrucciones sobre cómo ejecutar los procesos del servidor para el alojamiento de sesiones de juego. El GameLift agente

de Amazon implementa estas instrucciones en cada grupo de contenedores de réplicas de la flota.

La configuración de tiempo de ejecución de una flota determina cuántos procesos de servidor se ejecutan simultáneamente en cada grupo de contenedores de réplicas. Esta configuración afecta a la forma en que calcula los límites de recursos de su grupo de contenedores y a la forma en que elige un tipo de instancia para su flota. Debe equilibrar estos tres elementos a la hora de diseñar su flota.

Para obtener más información sobre cómo utilizar las configuraciones de tiempo de ejecución, consulteAdministración de la forma en que Amazon GameLift lanza los servidores de juegos.

Establezca otros ajustes de flota opcionales

Puede utilizar las siguientes funciones opcionales al configurar una flota de contenedores:

- Configura tus servidores de juegos para acceder a otros AWS recursos. Consulte Comunicación con otros recursos de AWS de sus flotas.
- Evita que las sesiones de juego con jugadores activos finalicen prematuramente durante un evento a escala reducida.
- Limita el número de sesiones de juego que una persona puede crear en la flota en un periodo de tiempo limitado.

Crea definiciones de grupos de contenedores para una flota de GameLift contenedores de Amazon

Esta documentación corresponde a una función que se encuentra en una versión preliminar pública. Está sujeta a cambios.

La definición de un grupo de contenedores describe cómo implementar sus aplicaciones de servidor de juegos en contenedores en una flota de contenedores. Se trata de un plano que identifica el conjunto de contenedores que se utilizarán en la flota y cómo se gestionarán. Al crear una flota de contenedores, se especifican las definiciones de grupos de contenedores que se van a implementar en la flota. Para obtener más información sobre los grupos de contenedores, consulteComponentes de la flota de contenedores.

Antes de comenzar

Realice los siguientes pasos:

- Diseñe una arquitectura de contenedores para alojar sus servidores de juegos. Consulte Diseña una flota de GameLift contenedores de Amazon.
- Planifique las definiciones de contenedores para incluirlas en el grupo de contenedores. Si utilizas la AWS CLI, crea tu definición de contenedor en un archivo JSON.
- Coloca las imágenes finales del contenedor en un registro de Amazon Elastic Container Registry (Amazon ECR) en el Región de AWS mismo lugar en el que planeas crear el grupo de contenedores. Amazon GameLift almacena una instantánea de cada imagen al crear la definición del grupo de contenedores y utiliza la copia al desplegarla en una flota de contenedores. Consulte Prepara una imagen de contenedor con el software de tu servidor de juegos.
- Compruebe que el AWS usuario tiene permisos de IAM para acceder al repositorio de Amazon ECR. Consulte Administrar los permisos de usuario para Amazon GameLift. Como mínimo, necesita permisos para las siguientes acciones:
  - `ecr:DescribeImages`
  - `ecr:BatchGetImage`
  - `ecr:GetDownloadUrlForLayer`

Clonar una definición de grupo de contenedores

Puedes usar la GameLift consola de Amazon para clonar una definición de grupo de contenedores existente.

Para clonar un grupo de contenedores

1. En la GameLift consola de Amazon, ve al panel de navegación izquierdo y selecciona Grupos de contenedores.
2. En la página de lista de grupos de contenedores, selecciona el grupo de contenedores existente que deseas clonar. Tras seleccionar un grupo de contenedores, el botón Clonar estará activo.
3. Elija Clonar. Esta acción abre el asistente de creación de grupos de contenedores con los ajustes predefinidos.
4. Introduzca un nombre nuevo para el grupo de contenedores clonado. El grupo de contenedores de la misma región debe tener nombres únicos.
5. Recorra las páginas del grupo de contenedores y de definición de contenedores, revise y cree el nuevo grupo de contenedores.

Cree una réplica de la definición de grupo de contenedores

Un grupo de contenedores de réplicas administra el software del servidor de juegos. Un grupo de contenedores de réplicas tiene al menos un contenedor que ejecuta los procesos de Amazon GameLift Agent y del servidor de juegos. Es posible que el grupo tenga contenedores «sidecar» adicionales para ejecutar el software de soporte.

En este tema se describe cómo crear una definición de grupo de contenedores mediante la GameLift consola de Amazon o las herramientas AWS CLI. Para obtener información más detallada sobre cómo configurar los grupos de contenedores, consulteDiseña una flota de GameLift contenedores de Amazon.

Console

En la GameLift consola de Amazon, selecciona la Región de AWS ubicación en la que deseas crear el grupo de contenedores.

Abre la barra de navegación izquierda de la consola y selecciona Grupos de contenedores. En la página Grupos de contenedores, selecciona Crear grupo de contenedores.

Paso 1: Defina los detalles del grupo.

1. Introduzca un nombre de definición de grupo de contenedores. Este nombre debe ser exclusivo de la región Cuenta de AWS and. En la consola, las definiciones de los grupos se muestran por nombre, por lo que puede resultar útil asignar etiquetas significativas.
2. Seleccione la estrategia de programación de réplicas.
3. En Límite total de memoria, introduzca la memoria máxima disponible para el grupo de contenedores. Si necesita ayuda para calcular este valor, consulteEstablece límites de recursos.
4. En Límite total de CPU, introduzca la potencia de cálculo máxima disponible para el grupo de contenedores. Si necesita ayuda para calcular este valor, consulteEstablece límites de recursos.

Paso 2: Añadir definiciones de contenedores.

Defina el contenedor con su aplicación de servidor de juegos y el Amazon GameLift Agent. Este es tu contenedor de réplicas esencial.

1. Proporcione un nombre para la definición del contenedor. Cada contenedor definido para el grupo debe tener un valor de nombre único.
2. Identifique el URI de la imagen ECR de Amazon de la imagen del contenedor. Introduzca cualquiera de los siguientes formatos:

   - Solo URI de imagen: `[Cuenta de AWS].dkr.ecr.[Región de AWS].amazonaws.com/[repository ID]`
   - URI de imagen + resumen: `[Cuenta de AWS].dkr.ecr.[Región de AWS].amazonaws.com/[repository ID]@[digest]`
   - URI de imagen + etiqueta: `[Cuenta de AWS].dkr.ecr.[Región de AWS].amazonaws.com/[repository ID]:[tag]`
3. En el caso del contenedor esencial, se selecciona automáticamente Sí para la primera definición del contenedor. Si añade otra definición de contenedor, puede activar o desactivar esta configuración para cada definición. Para obtener más información, consulte Designe los contenedores esenciales.
4. Establezca uno o más rangos de puertos de contenedores internos. Este contenedor aloja tus servidores de juegos, así que define un rango con suficientes puertos para que cada proceso del servidor se ejecute en el grupo de contenedores. Para obtener más información, consulte Configure las conexiones de red.
5. Los ajustes opcionales, las anulaciones y las variables de entorno te permiten especificar los valores que se transferirán al contenedor en el momento del lanzamiento. Los valores que establezca aquí anulan cualquier configuración que ya esté en la imagen del contenedor.
6. Establezca límites de contenedor opcionales para administrar la asignación de recursos para este contenedor. Para obtener más información, consulte Establece límites de recursos.
7. Defina contenedores no esenciales adicionales según sea necesario:

   - Proporcione un nombre de definición de contenedor y un URI de imagen ECR. Los contenedores no esenciales no deben funcionar con Amazon GameLift Agent.
   - Establezca un rango de puertos de contenedores interno solo si los contenedores tienen procesos que necesiten acceso a la red.
   - Si lo desea, configure un Health Check para el contenedor. Cuando un contenedor no esencial no pasa una comprobación de estado, solo se solicita que se reinicie el contenedor defectuoso.
   - Si lo desea, defina las anulaciones, las variables de entorno y los límites de asignación de recursos según sea necesario.

Paso 3: Configurar las dependencias.

Si tiene más de un contenedor en su definición de grupo de contenedores, puede definir las dependencias entre ellos. Utilice las dependencias para configurar las secuencias de inicio y cierre en función del estado del contenedor. Para obtener más información, consulte Establezca las dependencias de los contenedores.

1. Identifica el nombre del contenedor para el que quieres añadir una dependencia. Este contenedor no se inicia hasta que se cumpla la condición de dependencia.
2. Identifique el nombre y la condición del contenedor de la dependencia. Este contenedor debe cumplir la condición antes de que el contenedor dependiente pueda comenzar.
3. Establezca dependencias adicionales según sea necesario. Puede crear varias dependencias para cualquier contenedor. Evita crear dependencias circulares.

Paso 4: Revisa y crea.

1. Revise todos los ajustes de definición de sus grupos de contenedores. No puede cambiar la configuración de una definición de grupo de contenedores una vez creada. Usa Editar para realizar cambios en cualquier sección, incluida cada una de las definiciones de contenedor del grupo.
2. Cuando termines de revisarla, selecciona Crear.

   Si la solicitud se ha realizado correctamente, la consola mostrará la página de detalles del nuevo recurso de definición de grupos de contenedores. Inicialmente`COPYING`, el estado es el siguiente: Amazon GameLift comienza a tomar instantáneas de todas las imágenes de contenedores del grupo. Cuando se complete esta fase, el estado de la definición del grupo de contenedores cambiará a`READY`. La definición de un grupo de contenedores debe estar en `READY` estado para poder crear una flota de contenedores con ella.

AWS CLI

Cuando utilice la AWS CLI para crear una definición de grupo de contenedores, mantenga las configuraciones de definición de contenedores en un `JSON` archivo independiente. Puede hacer referencia al archivo en el comando CLI. Consulte Crea un `JSON` archivo de definición de contenedor para ver ejemplos de esquemas.

Cree una definición de grupo de contenedores

Para crear una nueva definición de grupo de contenedores, utilice el comando `create-container-group-definition` CLI. Para obtener más información sobre este comando, consulte la Referencia create-container-group-definitionde comandos de la AWS CLI.

Example : Grupo de contenedores de réplicas

Este ejemplo ilustra una solicitud de definición de grupo de contenedores de réplicas. La estructura de comandos para crear las definiciones de réplicas y grupos de demonios es básicamente idéntica. Los detalles específicos de cada tipo de grupo se describen en las definiciones de los contenedores individuales.

En este ejemplo, se supone que has creado un archivo JSON con las definiciones de contenedor de este grupo.

```
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
```

Crea un **JSON** archivo de definición de contenedor

Al crear una definición de grupo de contenedores, también define los contenedores del grupo. Una definición de contenedor especifica el repositorio de Amazon ECR en el que se almacena la imagen del contenedor y las configuraciones opcionales para los puertos de red, los límites de uso de CPU y memoria y otros ajustes. Recomendamos crear un único `JSON` archivo con las configuraciones de todos los contenedores de un grupo de contenedores. Mantener un archivo es útil para almacenar, compartir y realizar un seguimiento de las versiones de estas configuraciones críticas. Si usa la AWS CLI para crear sus definiciones de grupos de contenedores, puede hacer referencia al archivo en el comando.

Para crear una definición de contenedor

1. Cree y abra un `.JSON` archivo nuevo. Por ejemplo:

```
[~/work/glc]$ vim SimpleServer.json
```

2. Cree una definición de contenedor independiente para cada uno de los contenedores del grupo. Copie el siguiente contenido de ejemplo y modifíquelo según sea necesario para sus contenedores. Para obtener más información sobre la sintaxis de una definición de contenedor, consulta ContainerDefinitionInputla referencia de la GameLift API de Amazon.
3. Guarde el archivo localmente para poder consultarlo en un comando AWS CLI.

Ejemplo: definición esencial de contenedor de réplicas

Example

En este ejemplo se describe el contenedor esencial para su grupo de contenedores de réplicas. El contenedor de réplicas esencial incluye tu aplicación de servidor de juegos, el Amazon GameLift Agent, y puede incluir otro software de apoyo para el alojamiento de tu juego. La definición debe incluir un nombre, un URI de imagen y una configuración de puertos. Este ejemplo también establece algunos límites de recursos específicos del contenedor.

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

Crea una flota de GameLift contenedores de Amazon

Esta documentación corresponde a una función que se encuentra en una versión preliminar pública. Está sujeta a cambios.

Cuando hayas creado las definiciones de tus grupos de contenedores, usa la GameLift consola de Amazon o AWS Command Line Interface (AWS CLI) para crear una flota de contenedores.

Tras crear una flota nueva, el estado de la flota pasa por varias etapas a medida que Amazon GameLift despliega tus grupos de contenedores en cada instancia de la flota e inicia los servidores del juego. Cuando la flota alcance su estado`ACTIVE`, estará lista para albergar sesiones de juego. Para obtener ayuda sobre problemas de creación de flotas, consulte Solución de problemas con la flota de Amazon GameLift.

Console

En la GameLift consola de Amazon, selecciona la Región de AWS ubicación en la que quieres crear la flota. Las definiciones de los grupos de contenedores deben estar en la misma región en la que deseas crear la flota.

Abre la barra de navegación izquierda de la consola y selecciona Flotas. En la página Flotas, selecciona Crear flota.

Paso 1: elige el tipo de cómputo

- Elija el tipo de cómputo de Containers.

Paso 2: Defina los detalles de la flota

1. En la sección de detalles de la flota, introduce el nombre y la descripción de la flota.
2. En la sección de detalles del grupo de contenedores, identifique los grupos de contenedores que desee desplegar en la flota. Debe añadir un grupo de contenedores de réplicas. Si lo desea, puede añadir un grupo de contenedores daemon. Cada grupo debe estar en estado`READY`.
3. Establezca el rango de puertos de conexión para la flota. Para obtener más información, consulte Configure las conexiones de red.
4. Si lo desea, especifique las réplicas que desee implementar por instancia. Puedes especificar el número deseado o puedes dejar que Amazon GameLift calcule el número máximo posible. Si especificas un número deseado superior al máximo calculado, no se podrá crear la flota.

No puedes cambiar esta configuración una vez creada la flota. Para obtener más información sobre el embalaje grupal de réplicas de contenedores, consulteConceptos clave.

5. (Opcional) En Detalles adicionales:

   a. En el caso del rol de instancia, especifica un rol de IAM que autorice a las aplicaciones de la versión de tu juego a acceder a otros AWS recursos de tu cuenta. Para obtener más información, consulte Comunicación con otros recursos de AWS de sus flotas. Para crear una flota con un rol de instancia, la cuenta debe tener el permiso `PassRole` de IAM. Para obtener más información, consulte Ejemplos de permisos de IAM para Amazon GameLift.

   b. En Grupo de métricas, introduzca el nombre de un grupo de métricas de flota nuevo o existente. Puede agregar las métricas de varias flotas añadiéndolas al mismo grupo de métricas.

Paso 3: Defina los detalles de la instancia

1. En el despliegue de instancias, seleccione una o más ubicaciones remotas para implementar las instancias. La región de origen se selecciona automáticamente (es la región en la que va a crear la flota). Si selecciona ubicaciones adicionales, las instancias de flota también se implementarán en estas ubicaciones.

   > **Important**
   >
   > Para usar las regiones que no están habilitadas de forma predeterminada, habilítalas en tu Cuenta de AWS.
   >
   > - Las flotas con regiones que no están habilitadas, y que haya creado antes del 28 de febrero de 2022, no se verán afectadas.
   > - Para crear nuevas flotas con varias ubicaciones o actualizar las existentes, habilite primero las regiones que desee utilizar.
   >
   > Para obtener más información sobre las regiones que no están habilitadas de forma predeterminada y cómo habilitarlas, consulte Administración de Regiones de AWS en la Referencia general de AWS.

2. Seleccione una configuración de instancia para la flota. La consola calcula automáticamente la vCPU y la memoria mínimas requeridas (en función de los límites totales que establezca

para cada grupo de contenedores). Filtra la lista completa de tipos de instancias disponibles en función de los requisitos de recursos y de las ubicaciones que hayas introducido. Puede añadir filtros adicionales según sea necesario.

Para obtener más información sobre cómo elegir un tipo de instancia, consulte Configure una flota de contenedores. El tamaño del tipo de instancia que elija afectará a la forma en que se empaquetan los grupos de contenedores de réplicas en cada instancia de la flota. En función de su elección, considere la posibilidad de revisar la configuración para ver las réplicas deseadas por instancia.

Paso 4: Configurar el tiempo de ejecución

La configuración del tiempo de ejecución determina cómo se inician y ejecutan los procesos del servidor del juego. Estas instrucciones se pasan al Amazon GameLift Agent, que las implementa de la misma manera en cada grupo de contenedores de réplicas. Puede actualizar la configuración de tiempo de ejecución de una flota llamando UpdateRuntimeConfiguration.

1. En Ruta de lanzamiento, introduce la ruta a un ejecutable del juego.
2. En Parámetros de lanzamiento (opcional), introduzca la información para pasarla al archivo ejecutable del juego como un conjunto de parámetros de línea de comandos.
3. Especifique el número de procesos simultáneos que se van a mantener en ejecución en cada grupo de contenedores de réplicas. Revisa las GameLift cuotas de Amazon en cuanto al número de procesos de servidor por instancia. Las restricciones a los procesos del servidor simultáneos por instancia se aplican a todos los procesos simultáneos de todas las configuraciones. Si configura la flota de manera que se supere el límite, la flota no podrá activarse.
4. Establece límites opcionales para las activaciones simultáneas de las sesiones de juego. Estos ajustes te permiten limitar la cantidad de recursos que se consumen al iniciar una nueva sesión de juego. Las activaciones de las sesiones de juego pueden afectar al rendimiento de las sesiones de juego existentes.
5. Establezca la configuración del puerto EC2 para permitir que el tráfico externo acceda a los procesos que se ejecutan en la flota. Especifique algunos o todos los números de los puertos de conexión definidos para la flota. No tienes que configurar estos puertos al crear la flota, pero sin ellos el tráfico no podrá conectarse a los servidores de tus juegos. Para actualizar la configuración de los puertos de una flota más adelante, llama UpdateFleetPortSettings

6. En los ajustes de recursos de la sesión de juego, configura las siguientes funciones opcionales:

   a. Activa o desactiva la política de protección de escalado del juego. Con la protección activada, Amazon GameLift no cerrará las instancias durante un evento de reducción de escala si están organizando una sesión de juego activa.

   b. Establece un límite máximo de creación de recursos para limitar el número de sesiones de juego que un jugador puede crear durante un período de tiempo específico.

Paso 5: Configura las etiquetas

- Especifique los pares de clave y valor para añadir etiquetas a la compilación (opcional). Seleccione Siguiente para continuar con la revisión de la creación de la flota.

Paso 6: Revisar y crear.

- Revise los ajustes de configuración de su flota.

  Puede actualizar los metadatos de la flota y la configuración en cualquier momento, independientemente del estado de la flota. Para obtener más información, consulte Administración de las flotas de Amazon GameLift. Puede actualizar la capacidad de la flota después de que la flota haya alcanzado el estado ACTIVO. Para obtener más información, consulte Escalación de la capacidad de alojamiento de Amazon GameLift. También puede añadir ubicaciones remotas o eliminarlas.

  Cuando termines de revisarla, selecciona Crear.

  Si la solicitud se ha realizado correctamente, la consola mostrará la página de detalles del nuevo recurso de flota. InicialmenteNEW, el estado es: Amazon GameLift inicia el proceso de creación de la flota. Puede hacer un seguimiento del estado de la flota nueva en la página Flotas. Una flota está lista para albergar sesiones de juego cuando alcanza el estadoACTIVE.

AWS CLI

Para crear una flota de contenedores con el AWS CLI, abre una ventana de línea de comandos y usa el `create-fleet` comando. Para obtener más información sobre este comando, consulte `create-fleet`la Referencia de AWS CLI comandos.

El ejemplo de `create-fleet` solicitud que se muestra a continuación crea una nueva flota de contenedores con las siguientes características:

- ContainerGroupsConfiguration Especifica una definición de grupo de contenedores de réplica única:`MegaFrogRaceServer.NA.v2`. Se implementarán tres copias del grupo de réplicas en cada instancia de flota. Cada instancia tiene 30 puertos de conexión disponibles para acceder a los procesos de la instancia.
- La flota utiliza instancias c5.large On-Demand.
- Despliega grupos de contenedores en las siguientes ubicaciones:
  - us-west-2 (región de origen)
  - ca-central-1 (ubicación remota)
- Cada grupo de contenedores de réplicas de una instancia ejecutará 5 procesos del servidor de juegos de forma simultánea, lo que permitirá que cada instancia albergue hasta 15 sesiones de juego a la vez.
- En cada grupo de contenedores de réplicas, Amazon GameLift permite que se activen dos nuevas sesiones de juego al mismo tiempo. Asimismo, terminará cualquier sesión de juego de activación si no está lista para alojar jugadores en un plazo de 300 segundos.
- Todas las sesiones de juego alojadas en instancias en esta flota tienen la protección de sesión de juego activada.
- Los jugadores individuales pueden crear tres nuevas sesiones de juego en un periodo de 15 minutos.

```
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 solicitud de creación de flota se ha realizado correctamente, Amazon GameLift devuelve un conjunto de atributos de flota que incluye los ajustes de configuración que has solicitado y un nuevo identificador de flota. GameLift A continuación, Amazon establece el estado de la flota y los estados de ubicación en Nuevo e inicia el proceso de activación de la flota. Puede realizar un seguimiento del estado de la flota y ver más información sobre la flota con estos comandos de la CLI:

- describe-fleet-events
- describe-fleet-attributes
- describe-fleet-capacity
- describe-fleet-port-settings
- describe-fleet-utilization
- describe-runtime-configuration
- describe-fleet-location-attributes
- describe-fleet-location-capacity
- describe-fleet-location-utilization

Puede cambiar la capacidad de la flota y otras opciones de configuración según sea necesario mediante estos comandos:

- update-fleet-attributes
- update-fleet-capacity
- update-fleet-port-settings
- update-runtime-configuration
- create-fleet-locations
- delete-fleet-locations

Gestiona tus flotas de GameLift contenedores de Amazon

Esta documentación corresponde a una función que se encuentra en una versión preliminar pública. Está sujeta a cambios.

Cuando desee obtener información sobre su flota de contenedores o realizar cambios, puede utilizar las siguientes acciones para gestionar su flota de contenedores.

Vea los recursos de

A continuación, se muestran algunas formas de obtener información sobre los recursos de su flota de contenedores.

- DescribeCompute- Devuelve detalles sobre un contenedor registrado como cómputo.
- DescribeContainerGroupDefinition- Devuelve detalles sobre la definición de un grupo de contenedores. Este recurso describe cómo se configuran el grupo y sus contenedores.
- DescribeFleetAttributes- Obtiene los atributos de la flota, que incluyen el rango de puertos de conexión y otros atributos.
- DescribeFleetCapacity- Obtiene un recuento de los grupos de réplicas de contenedores de la flota y sus estados.
- DescribeRuntimeConfiguration- Describe los procesos de servidor que se ejecutan en cada grupo de contenedores de réplicas.
- GetComputeAccess- Proporciona acceso remoto a una instancia que aloja el grupo de contenedores.
- GetComputeAuthToken- Solicita un token de autenticación a Amazon GameLift para un recurso informático de una flota de contenedores.
- ListCompute- Muestra los grupos de contenedores registrados como ordenadores.
- ListContainerGroupDefinitions- Muestra las definiciones de los grupos de contenedores.
- ListFleets- Muestra las flotas que utilizan un grupo de contenedores específico.

Actualizar recursos

A continuación se muestran algunas formas de crear y modificar los recursos de una flota de contenedores.

- CreateContainerGroupDefinition- Crea una definición de grupo de contenedores.

- CreateFleet- Crea una flota de contenedores cuando `ComputeType` está configurado en`CONTAINER`.
- RegisterCompute- Registra los cálculos con una flota de contenedores.
- UpdateFleetAttributes- Actualiza los atributos mutables de una flota, como las opciones de configuración de la flota de Anywhere.
- UpdateFleetCapacity- Actualiza la configuración de capacidad de una flota gestionada por EC2 o de contenedores.
- UpdateRuntimeConfiguration- Actualiza la configuración del tiempo de ejecución, que describe los procesos del servidor que se deben ejecutar en cada grupo de contenedores de réplicas registrado como informático.

Eliminación de recursos

A continuación, se muestran algunas formas de eliminar los recursos de la flota de contenedores.

- DeleteContainerGroupDefinition- Elimina la definición de un grupo de contenedores.
- DeleteFleet- Elimina una flota.
- DeregisterCompute- Elimina un recurso informático de una flota de contenedores.

Ampliación de las flotas de GameLift contenedores de Amazon

Esta documentación corresponde a una función que se encuentra en una versión preliminar pública. Está sujeta a cambios.

Una de las tareas más complicadas del alojamiento de juegos es escalar la capacidad para satisfacer la demanda de los jugadores sin malgastar dinero en recursos innecesarios. En una flota de contenedores, se amplía la capacidad de la flota añadiendo o quitando instancias de flota.

Cuando creas una nueva flota, Amazon GameLift establece la capacidad deseada de la flota en una instancia y despliega una instancia en la región de origen de la flota. Para una flota de múltiples ubicaciones, Amazon GameLift despliega una instancia en la región de origen y en cada ubicación remota. Una vez alcanzado el estado de la flota`ACTIVE`, puede aumentar la capacidad deseada para ampliarla o reducirla para reducirla.

Puedes usar las funciones de GameLift escalado de Amazon para cambiar la capacidad manualmente o configurar el escalado automático en función de la demanda de los jugadores:

- Configura el escalado automático con el seguimiento de objetivos. Consulte Escalado automático basado en objetivos.
- Cambia manualmente la capacidad de tu flota. Consulte Configuración manual de la capacidad de una flota de Amazon GameLift.

Al ampliar una flota de contenedores, ten en cuenta cómo la adición o eliminación de instancias afecta a la capacidad de la flota para albergar sesiones de juego y jugadores.

- Sesiones de juego por instancia
  - Cada proceso del servidor de juegos que se ejecuta en una instancia representa la capacidad de alojar una sesión de juego.
  - Usa esta fórmula para calcular el número de sesiones de juego que se ejecutan simultáneamente en una instancia de flota de contenedores:

    ```
    [Game sessions per instance] = [# of processes per replica container group] * [# of
     replica container groups per instance]
    ```

    - Para los procesos por grupo de contenedores de réplicas, llama DescribeRuntimeConfigurationy cuenta el número de ejecuciones simultáneas de los procesos del servidor de juegos.
    - Para grupos de contenedores de réplicas por instancia, llama DescribeFleetAttributespara obtener el `DesiredReplicaContainerGroupPerInstance` valor. Si este valor no está establecido, `MaxReplicaContainerGroupsPerInstance` utilícelo.
- Jugadores por instancia
  - Tú decides el número de espacios para jugadores que quieres permitir en cada sesión de juego. En función de cómo gestione tu solución de alojamiento la ubicación de las sesiones de juego, puedes definir los jugadores por sesión de juego en tu configuración de matchmaking o en tus llamadas para iniciar la ubicación de una sesión de juego.
  - Usa esta fórmula para calcular el número de jugadores que pueden jugar a tu juego simultáneamente en una instancia de flota de contenedores:

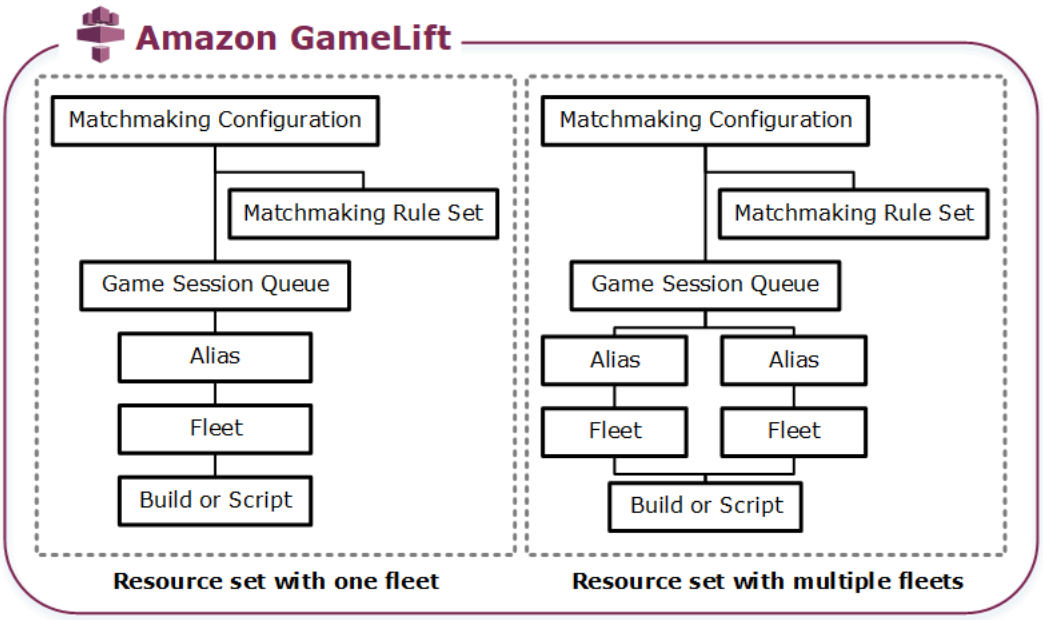
    ```
    [Players per instance] = [# of game sessions per instance] * [# of player slots per
     game session]
    ```

Para obtener la capacidad total actual de una flota de contenedores, llama a DescribeFleetCapacityo DescribeFleetLocation Capacity para obtener el número de réplicas de grupos de contenedores de la flota. Los grupos activos son aquellos que actualmente organizan sesiones de juego. Los grupos inactivos están preparados para organizar una nueva sesión de juego. Multiplique estos valores por el número de procesos de servidor por grupo de contenedores de réplicas.

Administración de recursos de alojamiento de Amazon GameLift.

En esta sección se proporciona información detallada sobre cómo configurar los recursos administrados de Amazon GameLift para ejecutar los servidores de juegos y alojar sesiones de juego para los jugadores. Debe configurar e implementar los recursos, escalar la capacidad para satisfacer las necesidades de los jugadores y localizar los recursos disponibles para organizar las sesiones de juego.

En el diagrama siguiente, se muestra el modo en que los objetos de recursos de Amazon GameLift se relacionan entre sí. Utilice una compilación o un script para crear una flota, asignar un alias a una flota y añadir flotas a la cola de una sesión de juego mediante su alias. Para los juegos que utilizan el emparejamiento de FlexMatch, utilice la cola de sesiones del juego y un conjunto de reglas de emparejamiento para crear una configuración de emparejamiento.



Código del servidor de juegos

- Compilación: el software del servidor de juegos personalizado que se ejecuta en Amazon GameLift y que aloja sesiones de juego para sus jugadores. Una compilación del juego representa el conjunto de archivos que ejecuta el servidor de juegos en un sistema operativo determinado y que debe integrar con Amazon GameLift. Suba los archivos de compilación de juegos a Amazon GameLift en las Regiones de AWS, donde planea configurar las flotas.

Para obtener más información, consulte Carga de una compilación del servidor de juegos personalizada en Amazon GameLift.

- Script: su configuración y lógica de juego personalizada para utilizarla con Realtime Servers. Configure Realtime Servers para sus clientes de juegos mediante la creación de un script con JavaScript y añada una lógica de juego personalizada para alojar sesiones de juego para sus jugadores. Para obtener más información, consulte Carga de un script de Realtime Servers en Amazon GameLift.

Flota

Un conjunto de recursos informáticos que ejecutan los servidores de juegos y alojan las sesiones de juego para los jugadores. Para obtener información sobre dónde puede implementar flotas, consulte Ubicaciones GameLift de alojamiento de Amazon. Para obtener información sobre la creación de flotas, consulte Configuración de las flotas de Amazon GameLift.

Alias

Un identificador abstracto de una flota que puede utilizar para cambiar la flota a la que están conectados los jugadores en cualquier momento. Para obtener más información, consulte Añadir un alias a una GameLift flota de Amazon.

Cola de sesiones de juego

Un mecanismo de ubicación de las sesiones de juego que recibe las solicitudes de nuevas sesiones de juego y busca los servidores de juego disponibles para alojar las nuevas sesiones. Para obtener más información sobre las colas de sesiones, consulte Configuración de colas de Amazon GameLift para la ubicación de las sesiones de juego.

Carga de compilaciones y scripts en Amazon GameLift

Antes de implementar sus servidores de juegos multijugador para alojarlos con Amazon GameLift, debe cargar los archivos del servidor de juegos. En los temas de esta sección se proporcionan instrucciones sobre cómo preparar y cargar archivos de compilación o archivos de script de servidor de Realtime para un servidor de juegos personalizado.

Temas

- Carga de una compilación del servidor de juegos personalizada en Amazon GameLift
- Carga de un script de Realtime Servers en Amazon GameLift

Carga de una compilación del servidor de juegos personalizada en Amazon GameLift

Después de integrar el servidor de juegos con Amazon GameLift, cargue los archivos de compilación en Amazon GameLift. En este tema se explica cómo empaquetar los archivos de compilación del juego, crear un script de instalación de la compilación opcional y, a continuación, cargar los archivos mediante AWS Command Line Interface (AWS CLI) o un SDK de AWS.

Temas

Empaquetado de los archivos de compilación del juego

Antes de cargar el servidor de juegos configurado en Amazon GameLift, empaquete los archivos de compilación del juego en un directorio de compilación. Este directorio debe incluir todos los componentes necesarios para ejecutar los servidores de juegos y las sesiones de juego alojadas, entre los que se incluyen los siguientes:

- Archivos binarios del servidor de juegos: son los archivos binarios necesarios para ejecutar el servidor de juegos. Una compilación puede incluir archivos binarios para varios servidores de juegos diseñados para ejecutarlos en la misma plataforma. Para obtener una lista de plataformas admitidas, consulte Soporte de desarrollo con Amazon GameLift.
- Dependencias: cualquier archivo dependiente necesario para que los archivos ejecutables del servidor de juegos funcione. Por ejemplo, activos, archivos de configuración y bibliotecas dependientes.

> **Note**
>
> En el caso de compilaciones de juegos creadas con el SDK del servidor Amazon GameLift para C++ (incluidas las creadas con el complemento Unreal), incluya el DLL de OpenSSL para la misma versión de OpenSSL con la que creó el SDK del servidor. Consulte el archivo README del SDK del servidor para obtener más información.

- Script de instalación (opcional): archivo de script para administrar las tareas de instalación de la compilación del juego en los servidores de alojamiento de Amazon GameLift. Coloque ese archivo en la raíz del directorio de compilación. Amazon GameLift ejecuta el script de instalación como parte de la creación de la flota.

Puede configurar cualquier aplicación de la compilación, incluido el script de instalación, para obtener acceso seguro a los recursos de otros servicios de AWS. Para obtener información sobre cómo hacerlo, consulte Comunicación con otros recursos de AWS de sus flotas.

Después de empaquetar los archivos de compilación, asegúrese de que el servidor de juegos pueda ejecutarse en una instalación limpia de su sistema operativo de destino. De esa forma, se asegurará de que se incluyen todas las dependencias necesarias en el paquete y de que el script de instalación es correcto.

Creación de una compilación de Amazon GameLift

Al crear una compilación y cargar sus archivos, tiene dos opciones:

- Creación de una compilación a partir de un directorio de archivos. Esta es la opción más sencilla y la que se utiliza habitualmente.
- Cree una compilación con archivos en Amazon Simple Storage Service (Amazon S3). Con esta opción, puede administrar sus versiones de compilación en Amazon S3.

Con ambos métodos, Amazon GameLift crea un nuevo recurso de compilación con un ID de compilación único y otros metadatos. La compilación comienza con el estado Inicializado. Cuando Amazon GameLift adquiere los archivos del servidor de juegos, la compilación pasa al estado Listo.

Cuando la compilación esté lista, podrá implementarla en una nueva flota de Amazon GameLift. Para obtener más información, consulte Creación de una flota administrada por Amazon GameLift .Cuando Amazon GameLift configure la nueva flota, descargará los archivos de compilación en cada instancia de la flota e instalará los archivos de compilación.

Creación de una compilación a partir de un directorio de archivos

Para crear una compilación de juego almacenada en cualquier ubicación, incluido un directorio local, utilice el comando de la AWS CLI `upload-build`. Este comando crea un registro de compilación nuevo en Amazon GameLift y carga archivos desde la ubicación que especifique.

Envíe una solicitud de carga. En una ventana de línea de comandos, especifique el comando upload-build y los parámetros siguientes.

```
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: el entorno de tiempo de ejecución de la compilación del servidor de juegos. Debe especificar un valor para el sistema operativo. El valor no podrá actualizarse más tarde.
- server-sdk-version: la versión del SDK del servidor de Amazon GameLift con la que se ha integrado el servidor de juegos. Si no proporciona un valor, Amazon GameLift utiliza el valor predeterminado `4.0.2`. Si especifica una versión incorrecta del SDK del servidor, es posible que la compilación del servidor de juegos falle al llamar a `InitSdk` para establecer una conexión con el servicio de Amazon GameLift.
- build-root: es la ruta del directorio de los archivos de compilación.
- name: es un nombre descriptivo para la nueva compilación.
- build-version: los detalles de la versión de los archivos de compilación.
- region: la región de AWS en la que desea crear la compilación. Cree la compilación en la región en la que tiene previsto implementar las flotas. Si va a implementar el juego en varias regiones, cree una compilación en cada región.

> **Note**
>
> Vea su región predeterminada actual mediante el comando aws configure get region. Utilice el comando aws configure set region ***region name*** para cambiar la región predeterminada.

Ejemplos

```
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
```

Como respuesta a su solicitud de carga, Amazon GameLift proporciona el progreso de la carga. Si la carga se realiza correctamente, Amazon GameLift devuelve el nuevo ID de registro de compilación. El tiempo de carga depende del tamaño de los archivos del juego y de la velocidad de conexión.

Creación de una compilación con archivos en Amazon S3

Puede almacenar sus archivos de compilación en Amazon S3 y cargarlos en Amazon GameLift desde allí. Al crear la compilación, debe especificar la ubicación del bucket de S3 y Amazon GameLift recuperará los archivos de compilación directamente desde Amazon S3.

Para crear un recurso de compilación, realice el siguiente procedimiento:

1. Almacene los archivos de compilación en Amazon S.. Cree un archivo .zip que contenga los archivos de compilación empaquetados y cárguelos en un bucket de S3 en su Cuenta de AWS. Tome nota de la etiqueta del bucket y el nombre de archivo, ya que los necesitará para crear una compilación de Amazon GameLift.
2. Otorgue acceso a Amazon GameLift a los archivos de compilación. Para crear un rol de IAM;, siga las instrucciones en Acceso a un archivo de compilación de un juego en Amazon S3. En cuanto haya creado el rol, tome nota del nombre de recurso de Amazon (ARN) del nuevo rol, ya que lo necesitará para crear una compilación.
3. Cree una compilación. Utilice la consola de Amazon GameLift o la AWS CLI para crear un nuevo registro de compilación. Debe tener el permiso `PassRole`, tal y como se describe en Ejemplos de permisos de IAM para Amazon GameLift.

Console

1. En la consola de Amazon GameLift, en el panel de navegación, elija Alojamiento, Compilaciones.
2. En la página Compilaciones, seleccione Crear compilación.
3. En la página Crear compilación, en Configuración de compilación, realice el siguiente procedimiento:

   a. En Nombre, especifique un nombre de script.

   b. En Versión, escriba una versión. Como puede actualizar el contenido de una compilación, los datos de la versión pueden ayudarle a realizar un seguimiento de las actualizaciones.

   c. Para Sistema operativo (SO), elija el sistema operativo de la compilación del servidor de juegos. El valor no podrá actualizarse más tarde.

   d. En Compilación del servidor de juegos, especifique el URI de S3 del objeto de compilación que cargó en Amazon S3 y elija la versión del objeto. Si no recuerda el URI y la versión del objeto de Amazon S3, elija Explorar S3 y busque el objeto de compilación.

   e. Para el rol de IAM, elija el rol creado que permite a Amazon GameLift acceder a su bucket de S3 y al objeto de compilación.
4. En Etiquetas, añada etiquetas a la compilación introduciendo los pares Clave y Valor (opcional).
5. Seleccione Create (Crear).

Amazon GameLift asigna un ID a la nueva compilación y carga el archivo .zip designado. Puede ver la nueva compilación, incluido su estado, en la página Compilaciones

AWS CLI

Utilice el comando `create-build` para definir la nueva compilación y cargar los archivos de compilación del servidor.

1. Abra una ventana de línea de comandos y cambie a un directorio en el que pueda usar la AWS CLI.
2. Escriba el siguiente comando create-build:

```
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: es un nombre descriptivo para la nueva compilación.
- server-sdk-version: la versión del SDK del servidor de Amazon GameLift utilizado para integrar el servidor de juegos con Amazon GameLift. Si no proporciona un valor, Amazon GameLift utiliza el valor predeterminado `4.0.2`.
- operating-system: el entorno de tiempo de ejecución de la compilación del servidor de juegos. Debe especificar un valor para el sistema operativo. El valor no podrá actualizarse más tarde.
- build-version: los detalles de la versión de los archivos de compilación. Esa información puede resultar útil porque cada nueva versión del servidor de juegos requiere un nuevo recurso de compilación.
- storage-location
  - Bucket: nombre del bucket de S3 que contiene la compilación. Por ejemplo, "my_build_files".
  - Key: nombre del archivo .zip que contiene los archivos de compilación. Por ejemplo, "my_game_build_7.0.1, 7.0.2".
  - RoleARN: el ARN asignado al rol de IAM creado. Por ejemplo, "arn:aws:iam::111122223333:role/GameLiftAccess". Para ver una política de ejemplo, consulte Acceso a un archivo de compilación de un juego en Amazon S3.
- region: cree la compilación en la región de AWS en la que tiene previsto implementar las flotas. Si va a implementar el juego en varias regiones, cree una compilación en cada región.

> **Note**
>
> Le recomendamos que compruebe la región predeterminada actual mediante el comando `configure get`. Utilice el comando `configure set` para cambiar la región predeterminada.

Ejemplo

```
aws gamelift create-build \
    --operating-system WINDOWS_2016 \
    --storage-location
 "Bucket"="my_game_build_files","Key"="mygame_build_101.zip","RoleArn"="arn:aws:iam::111
gamelift" \
    --name "My Game Nightly Build" \
    --build-version "build 101" \
    --region us-west-2
```

3. Para ver la nueva compilación, utilice el comando `describe-build`.

Actualización de los archivos de compilación

Puede actualizar los metadatos de un recurso de compilación mediante la consola de Amazon GameLift o el comando `update-build` AWS CLI.

Una vez que haya creado una compilación de Amazon GameLift, no podrá actualizar los archivos de compilación asociados a ella. Para cada nuevo conjunto de archivos, cree una nueva compilación de Amazon GameLift. Con el comando `upload-build`, Amazon GameLift creará automáticamente un nuevo registro de compilación para cada solicitud. Si proporciona archivos de compilación mediante el comando `create-build`, cargue un nuevo archivo .zip de compilación con un nombre distinto en Amazon S3 y cree una compilación que haga referencia al nombre de archivo nuevo.

Tenga en cuenta estos consejos a la hora de implementar compilaciones actualizadas:

- Utilice colas e intercambie las flotas según sea necesario. Al configurar el cliente del juego con Amazon GameLift, especifique una cola en lugar de una flota. Con las colas, puede añadir nuevas flotas con la nueva compilación para la cola y eliminar las flotas antiguas. Para obtener más información, consulte Configuración de colas de Amazon GameLift para la ubicación de las sesiones de juego.
- Utilice alias para transferir jugadores a una compilación del juego nueva. Al integrar el cliente de juegos con Amazon GameLift, especifique un alias de flota en lugar de un ID de la flota. Para obtener más información, consulte Añadir un alias a una GameLift flota de Amazon.
- Configure actualizaciones de compilación automatizadas. Para ver ejemplos de scripts e información sobre cómo incorporar las implementaciones de Amazon GameLift a su sistema de

compilación, consulte Automatización de las implementaciones en Amazon GameLift en el blog de tecnología de juegos de AWS

Agregar un script de instalación de la compilación

Cree un script de instalación para el sistema operativo (SO) de la compilación del juego:

- Windows: cree un archivo de procesamiento por lotes denominado «`install.bat`».
- Linux: cree un archivo de script de shell denominado "`install.sh`".

Al crear un script de instalación, tenga en cuenta lo siguiente:

- El script no puede aceptar ninguna entrada por parte del usuario.
- Amazon GameLift instala la compilación y vuelve a crear los directorios de archivos del paquete de compilación en un servidor de alojamiento en las siguientes ubicaciones:
  - Flotas de Windows: `C:\game`
  - Flotas de Linux: `/local/game`
- Durante el proceso de instalación para flotas de Linux, el usuario que realiza la ejecución tiene acceso limitado a la estructura de archivos de la instancia. Dicho usuario dispone de acceso completo al directorio en el que se instalan los archivos de compilación. Si el script de instalación realiza acciones que requieren permisos de administrador, especifique el acceso de administrador mediante sudo. El usuario en ejecución para las flotas de Windows dispone de permisos de administrador de forma predeterminada. Los errores de permisos relacionados con el script de instalación generan un mensaje de evento que indica un problema con el script.
- En Linux, Amazon GameLift admite lenguajes comunes de intérprete de shell, como bash. Añada un shebang (como `#!/bin/bash`) al principio del script de instalación. Para verificar la compatibilidad con los comandos de shell que prefiera, inicie una sesión remota en una instancia de Linux activa y abra una solicitud del shell. Para obtener más información, consulte Conéctese remotamente a las instancias de GameLift la flota de Amazon.
- El script de instalación no puede confiar en una conexión de emparejamiento de VPC. La conexión de emparejamiento de VPC no estará disponible hasta que Amazon GameLift instale la compilación en las instancias de la flota.

Example Archivo bash de instalación de Windows

En este ejemplo, el archivo `install.bat` instala los componentes del tiempo de ejecución de Visual C++ necesarios para el servidor de juegos y escribe los resultados en un archivo de registro. El script incluye el archivo de componente en el paquete de compilación en la raíz.

```
vcredist_x64.exe /install /quiet /norestart /log c:\game\vcredist_2013_x64.log
```

Example Script de shell de instalación de Linux

Este archivo `install.sh` de ejemplo utiliza bash en el script de instalación y escribe los resultados en un archivo de registro.

```
#!/bin/bash
echo 'Hello World' > install.log
```

Este archivo `install.sh` de ejemplo muestra cómo puede utilizar el agente de Amazon CloudWatch para recopilar métricas personalizadas y de nivel del sistema, y gestionar la rotación de registros. Puesto que Amazon GameLift se ejecuta en una VPC de servicio, debe conceder permisos a Amazon GameLift para que asuma un rol AWS Identity and Access Management (IAM) en su nombre. Para permitir que Amazon GameLift asuma un rol, cree un rol que incluya la política administrada de AWS y utilice ese rol cuando cree una flota.

```
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
```

Carga de un script de Realtime Servers en Amazon GameLift

Cuando esté listo para implementar Realtime Servers en su juego, cargue los archivos de script de Realtime Server completos en Amazon GameLift. Para ello, cree un recurso de script de Amazon GameLift y especifique la ubicación de los archivos de script. También puede actualizar los archivos de scripts del servidor que ya estén implementados cargando nuevos archivos para un recurso de script existente.

Al crear un recurso de script nuevo, Amazon GameLift le asigna un ID de script único (por ejemplo, `script-1111aaaa-22bb-33cc-44dd-5555eeee66ff`) y carga una copia de los archivos de script. El tiempo de carga depende del tamaño de los archivos de script y de la velocidad de conexión.

Tras crear el recurso de script, Amazon GameLift implementa el script con una nueva flota de Realtime Servers. Amazon GameLift instala el script del servidor en cada instancia de la flota y coloca los archivos del script en `/local/game`.

Para solucionar los problemas de activación de flotas relacionados con el script del servidor, consulte Solución de problemas con la flota de Amazon GameLift.

Agrupamiento de los archivos de script

El script del servidor puede incluir uno o más archivos combinados en un único archivo.zip para cargarlos. El archivo.zip debe contener todos los archivos que el script necesita para ejecutarse.

Puede almacenar los scripts comprimidos en un directorio de archivos local o en un bucket de Amazon Simple Storage Service (Amazon S3).

Carga de los archivos de script desde un directorio local

Si tiene los archivos de scripts almacenados localmente, puede subirlos a Amazon GameLift desde ahí. Para crear el recurso de script, utilice la consola de Amazon GameLift o la AWS Command Line Interface (AWS CLI).

Amazon GameLift console

Para crear un recurso de script, realice el siguiente procedimiento:

1. Abra la consola de Amazon GameLift.
2. En el panel de navegación, elija Alojamiento, Scripts.
3. En la página Scripts, elija Crear script.
4. En la página Crear script, en Configuración de script, realice el siguiente procedimiento:
    a. En Nombre, especifique un nombre de script.
    b. En Nombre de la versión, escriba la información de la versión (opcional). Como puede actualizar el contenido de un script, los datos de la versión pueden resultar útiles para realizar un seguimiento de las actualizaciones.
    c. En Origen de script, seleccione Cargar un archivo .zip.
    d. Para Archivos de guiones, seleccione Elegir archivo, busque el archivo.zip que contiene el script y, a continuación, elija ese archivo.
5. En Etiquetas, añada etiquetas al script introduciendo los pares Clave y Valor (opcional).

6. Seleccione Create (Crear).

   Amazon GameLift asigna un ID al nuevo script y carga el archivo .zip designado. Puede ver el nuevo script, incluido su estado, en la página Scripts.

AWS CLI

Utilice el comando `create-script` AWS CLI para definir el nuevo script y cargar los archivos de script del servidor.

Para crear un recurso de script, realice el siguiente procedimiento:

1. Coloque el archivo.zip en un directorio en el que pueda utilizar la AWS CLI.
2. Abra una ventana de línea de comandos y cambie a un directorio en el que pueda colocar el archivo .zip.
3. Escriba el comando create-script y los parámetros siguientes. Para el parámetro --zip-file, asegúrese de añadir la cadena `fileb://` delante del nombre del archivo zip. Identifica el archivo como binario para que Amazon GameLift procese el contenido comprimido.

   ```
   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
   ```

   Ejemplo

   ```
   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
   ```

   Como respuesta a la solicitud, Amazon GameLift devuelve el objeto del nuevo script.

4. Para ver el nuevo script, llame a `describe-script`.

Carga de archivos de script desde Amazon S3

Puede almacenar sus archivos de script en un bucket de Amazon S3 y cargarlos en Amazon GameLift desde allí. Cuando cree el script, debe especificar la ubicación del bucket de S3 y Amazon GameLift recuperará los archivos de script desde S3.

Para crear un recurso de script, realice el siguiente procedimiento:

1. Almacene sus archivos de script en un bucket de S3. Cree un archivo.zip que contenga los archivos de script del servidor y cárguelo en un bucket de S3 en una Cuenta de AWS que controle. Tome nota del URI del objeto: lo necesitará para crear un script de Amazon GameLift.

   > **Note**
   >
   > Amazon GameLift no admite la carga de buckets de S3 con nombres que contienen un punto (.).

2. Otorgue acceso a Amazon GameLift a los archivos de script. Siga las instrucciones que se indican en Configurar un rol de servicio de IAM para Amazon GameLift para crear un rol de AWS Identity and Access Management (IAM) que permita a Amazon GameLift acceder al bucket de S3 que contiene el script del servidor. Después de crear el nuevo rol, tome nota del nombre, ya que lo necesitará al crear un script.
3. Cree un script. Utilice la consola de Amazon GameLift o la AWS CLI para crear un nuevo registro de script. Para realizar esta solicitud, debe tener el permiso `PassRole` de IAM, tal y como se describe en Ejemplos de permisos de IAM para Amazon GameLift.

Amazon GameLift console

1. En la consola de Amazon GameLift, en el panel de navegación, elija Alojamiento, Scripts.
2. En la página Scripts, elija Crear script.
3. En la página Crear script, en Configuración de script, realice el siguiente procedimiento:
   a. En Nombre, especifique un nombre de script.
   b. En Nombre de la versión, escriba la información de la versión (opcional). Como puede actualizar el contenido de un script, los datos de la versión pueden resultar útiles para realizar un seguimiento de las actualizaciones.
   c. En Origen de script, elija URI de Amazon S3.

d. Especifique el URI de S3 del objeto de script que cargó en Amazon S3 y, a continuación elija un valor para Versión de objeto. Si no recuerda el URI y la versión del objeto de Amazon S3, elija Explorar S3 y, a continuación, busque el objeto de script.

4. En Etiquetas, añada etiquetas al script introduciendo los pares Clave y Valor (opcional).
5. Seleccione Create (Crear).

   Amazon GameLift asigna un ID al nuevo script y carga el archivo .zip designado. Puede ver el nuevo script, incluido su estado, en la página Scripts.

AWS CLI

Utilice el comando `create-script` AWS CLI para definir el nuevo script y cargar los archivos de script del servidor.

1. Abra una ventana de línea de comandos y cambie a un directorio en el que pueda usar la AWS CLI.
2. Escriba el comando create-script y los parámetros siguientes. El parámetro --storage-location especifica la ubicación del bucket de Amazon S3 de los archivos 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
```

   Ejemplo

```
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
```

   Como respuesta a la solicitud, Amazon GameLift devuelve el objeto del nuevo script.

3. Para ver el nuevo script, llame a `describe-script`.

Actualización de archivos de script

Puede actualizar los metadatos de un recurso de script mediante la consola de Amazon GameLift o el comando `update-script` AWS CLI.

También puede actualizar el contenido del script de un recurso de script. Amazon GameLift implementa el contenido de los scripts en todas las instancias de la flota que utilizan el recurso de script actualizado. Cuando se implementa el script actualizado, las instancias lo utilizan al iniciar nuevas sesiones de juego. Las sesiones de juego que ya estaban en ejecución en el momento de la actualización no utilizan el script actualizado.

Para actualizar archivos de script, realice el siguiente procedimiento:

- En el caso de los archivos de scripts almacenados localmente, para cargar el archivo.zip del script actualizado, utilice la consola Amazon GameLift o el comando update-script.
- En el caso de los archivos de script almacenados en un bucket de Amazon S3, cargue los archivos de script actualizados en un bucket de Amazon S3. Amazon GameLift comprueba periódicamente si hay archivos de scripts actualizados y los recupera directamente del bucket de S3.

Configuración de las flotas de Amazon GameLift

En esta sección se proporciona información detallada sobre el diseño, la creación y el mantenimiento de flotas para su uso con Amazon GameLift. Puede utilizar las flotas de Amazon GameLift para implementar servidores de juegos personalizados y Realtime Servers.

Una flota representa sus recursos de alojamiento como un conjunto de instancias de Amazon Elastic Compute Cloud (Amazon EC2) o hardware físico. La ubicación de una flota determina dónde se implementan las instancias o el hardware para alojar las sesiones de juego de los jugadores. El tamaño de una flota y el número de sesiones de juego y jugadores a los que puede ofrecer soporte dependen del número de instancias o de la cantidad de hardware que le proporcione. Puede ajustar las instancias virtuales manualmente o mediante el escalado automático.

Muchos juegos en producción requieren más de una flota. Puede utilizar varias flotas, por ejemplo, para que más de una versión del servidor de juegos se ejecute simultáneamente, para proporcionar capacidad de respaldo a las flotas de spot o para incorporar redundancia.

Para saber cómo crear flotas diseñadas para las necesidades de su juego, consulte Guía de diseño de flotas de Amazon GameLift. Cuando la flota se esté ejecutando, consulte Escalación de la

capacidad de alojamiento de Amazon GameLift, Añadir un alias a una GameLift flota de Amazon y Configuración de colas de Amazon GameLift para la ubicación de las sesiones de juego.

Temas

Guía de diseño de flotas de Amazon GameLift

Esta guía de diseño describe las prácticas recomendadas para crear una flota de recursos de alojamiento para su uso con Amazon GameLift. Seleccione una combinación de recursos de alojamiento y conozca cómo configurarlos para que se adapten a su juego.

Temas

Elección de los recursos GameLift informáticos de Amazon

Para implementar sus servidores de juegos y alojar sesiones de juego para sus jugadores, Amazon GameLift utiliza los recursos de Amazon Elastic Compute Cloud (Amazon EC2) denominados instancias o su hardware físico. Al configurar una nueva flota mediante instancias, decida qué tipo de instancia necesita y cómo ejecutar los procesos del servidor de juegos en ellas. Cuando una flota de EC2 gestionada esté activa y lista para albergar sesiones de juego, puede añadir o eliminar instancias según sea necesario para adaptarla a la demanda de los jugadores.

Puedes implementar tus servidores de GameLift juegos de Amazon en una combinación de dos tipos de procesamiento:

- EC2 administrado: las flotas de EC2 administradas utilizan instancias de Amazon EC2 para alojar los servidores de juegos. Amazon GameLift gestiona las instancias y elimina la carga de gestionar el hardware y el software que supone alojar tus juegos.
- Amazon GameLift Anywhere: GameLift Anywhere las flotas de Amazon utilizan tu infraestructura existente para alojar los servidores de juegos, mientras que Amazon GameLift gestiona tus partidas y tus colas.

Al elegir los recursos informáticos de su flota, tenga en cuenta los factores siguientes:

- Hardware disponible
- Ubicación de la flota
- Instancias bajo demanda frente a instancias de spot
- Sistemas operativos
- Tipos de instancias
- Service Quotas

Hardware disponible

Tenga en cuenta la infraestructura existente en su implementación. Mientras migras los juegos a Amazon GameLift, puedes seguir utilizando tu infraestructura. Con Amazon GameLift Anywhere, puede utilizar su propia infraestructura junto con las instancias EC2 GameLift gestionadas por Amazon. También puedes usar tu infraestructura actual para alojar juegos más cerca de tus jugadores de lo que permiten GameLift las ubicaciones de Amazon compatibles. Para obtener más información sobre la configuración de GameLift Anywhere las flotas de Amazon, consulteCrea una GameLift Anywhere flota de Amazon.

Ubicación de la flota

Tenga en cuenta las ubicaciones geográficas en las que planea implementar sus servidores de juegos. La disponibilidad del tipo de instancia varía según Región de AWS la zona local.

En el caso de las flotas con varias ubicaciones, la disponibilidad y las cuotas de las instancias dependen de una combinación de la región de origen de la flota y las ubicaciones remotas seleccionadas. Para obtener más información sobre las ubicaciones de la flota, consulte Ubicaciones GameLift de alojamiento de Amazon.

En el caso de GameLift Anywhere las flotas de Amazon, usted determina la ubicación de su hardware físico. Para obtener más información sobre las ubicaciones personalizadas, consulte Amazon GameLift Anywhere.

Instancias bajo demanda frente a instancias de spot

Las instancias bajo demanda y las instancias de spot de Amazon EC2 ofrecen el mismo hardware y rendimiento, pero difieren en cuanto a disponibilidad y costo.

instancias bajo demanda

Puede adquirir una instancia bajo demanda cuando la necesite y mantenerla todo el tiempo que desee. Las instancias bajo demanda tienen un costo fijo, lo que significa que paga por la cantidad de tiempo que las utilice y no se genera ningún compromisos a largo plazo.

Spot Instances

Las instancias puntuales pueden ofrecer una alternativa rentable a las instancias bajo demanda al utilizar la AWS capacidad informática no utilizada. Los precios de las instancias puntuales fluctúan en función de la oferta y la demanda de cada tipo de instancia en cada ubicación. AWS puede interrumpir las instancias puntuales siempre que necesite recuperar la capacidad. Amazon GameLift utiliza colas y el algoritmo FleetiQ para determinar si AWS va a interrumpir una instancia puntual, y coloca la instancia en estado de reciclaje. Luego, cuando no hay sesiones de juego activas en la instancia, Amazon GameLift intenta reemplazarla.

Para obtener más información sobre cómo utilizar instancias de spot, consulte Usa instancias puntuales con Amazon GameLift.

Sistemas operativos

GameLift Las instancias de Amazon admiten compilaciones de servidores de juegos que se ejecutan en Microsoft Windows o Amazon Linux. Cuando subas una versión de juego a Amazon GameLift, especifica el sistema operativo del juego. Cuando creas una flota de Amazon EC2 para implementar la versión del juego, Amazon configura GameLift automáticamente las instancias con el sistema operativo de la compilación. Para obtener más información sobre los sistemas operativos del servidor de juegos compatibles, consulte Soporte de desarrollo con Amazon GameLift.

Al usar una GameLift Anywhere flota de Amazon, puedes usar cualquier sistema operativo compatible con tu hardware. GameLift AnywhereLas flotas de Amazon requieren que despliegues la versión del juego en el hardware y, al mismo tiempo, utilices Amazon GameLift para gestionar tus recursos en un solo lugar.

Tipos de instancias

El tipo de instancia de una flota de Amazon Ec2 determina la clase de hardware que utilizarán las instancias. Los tipos de instancia distintos ofrecen diferentes combinaciones de potencia informática, memoria, almacenamiento y funciones de red.

Al elegir entre los tipos de instancias disponibles para su juego, tenga en cuenta los siguientes aspectos:

- La arquitectura de cómputo de tu servidor de juegos: x64 o Arm (AWS Graviton).

> **Note**
>
> Las instancias de Graviton Arm requieren un GameLift servidor Amazon basado en el sistema operativo Linux. Se requiere el SDK de servidor 5.1.1 o posterior para C++ y C#. Se requiere el SDK de servidor 5.1.1 o posterior para continuar. Estas instancias no out-of-the-box admiten la instalación de Mono en Amazon Linux 2023 (AL2023) o Amazon Linux 2 (AL2).

- Los requisitos informáticos, de memoria y de almacenamiento de la compilación del servidor de juegos.
- El número de procesos del servidor que desea ejecutar por instancia.

Si utiliza un tipo de instancia más grande, es posible que pueda ejecutar varios procesos de servidor en cada instancia. Esto puede reducir la cantidad de instancias necesarias para satisfacer la demanda de los jugadores.

Para obtener más información:

- Para obtener más información sobre los tipos de instancias, consulte Tipos de instancias de Amazon EC2.
- Para obtener más información sobre la ejecución de varios procesos por instancia, consulte Administración de la forma en que Amazon GameLift lanza los servidores de juegos.

Service Quotas

Para ver las cuotas de servicio predeterminadas de Amazon GameLift y las cuotas actuales de las tuyas Cuenta de AWS, haz lo siguiente:

- Para obtener información general sobre las cuotas de servicio de Amazon GameLift, consulta los GameLiftpuntos de destino y las cuotas de Amazon en. Referencia general de AWS
- Para obtener una lista de los tipos de instancias disponibles por ubicación para tu cuenta, abre la página de cuotas de servicio de la GameLift consola de Amazon. Esta página también muestra el uso actual de su cuenta para cada tipo de instancia en cada ubicación.
- Para obtener una lista de las cuotas actuales de tu cuenta para los tipos de instancia por región, ejecuta el comando AWS Command Line Interface (AWS CLI) `describe-ec2-instance-limits`. Este comando devuelve el número de instancias activas de los que dispone en su región predeterminada (o en otra región que especifique).

Mientras te preparas para lanzar tu juego, completa un cuestionario de lanzamiento en la GameLift consola de Amazon. El GameLift equipo de Amazon utiliza el cuestionario de lanzamiento para determinar las cuotas y los límites correctos para tu juego.

Administración de la forma en que Amazon GameLift lanza los servidores de juegos

Puede establecer la configuración del tiempo de ejecución de una flota de EC2 administrada para que ejecute varios procesos de servidor de juegos por instancia. De esa manera, se utilizan sus recursos de alojamiento de manera más eficiente.

Modo en que una flota administra varios procesos

Amazon GameLift utiliza la configuración del entorno de ejecución de una flota para determinar el tipo y el número de procesos que ejecutar en cada instancia. Una configuración de tiempo de ejecución contiene al menos una configuración de proceso de servidor que representa un archivo ejecutable de servidor de juegos. Puede definir configuraciones de procesos de servidor adicionales para ejecutar otros tipos de procesos relacionados con el juego. Cada configuración del proceso del servidor contiene la siguiente información:

- El nombre de archivo y la ruta de acceso de un ejecutable de la compilación del juego.
- (Opcional) Los parámetros que se pasan al proceso del servidor durante el lanzamiento
- El número de procesos que se van a ejecutar al mismo tiempo.

Cuando se activa una instancia de la flota, lanza inmediatamente el conjunto de procesos del servidor definidos en la configuración del tiempo de ejecución. Con varios procesos, Amazon GameLift escalona el lanzamiento de cada proceso. Los procesos del servidor tienen una vida útil

limitada. Al finalizar, Amazon GameLift lanza nuevos procesos para mantener el número y el tipo de procesos del servidor definidos en la configuración del tiempo de ejecución.

Puede cambiar la configuración del entorno de ejecución de una flota en cualquier momento agregando, modificando o eliminando configuraciones de procesos del servidor. Cada instancia comprueba periódicamente si hay actualizaciones en la configuración del tiempo de ejecución de la flota para implementar los cambios. A continuación, le mostramos cómo Amazon GameLift realiza cambios en la configuración del tiempo de ejecución:

1. La instancia envía una solicitud a Amazon GameLift para obtener la versión más reciente de la configuración del tiempo de ejecución.
2. La instancia compara sus procesos activos con la configuración del tiempo de ejecución más reciente y, a continuación, realiza el siguiente procedimiento:
   - Si la configuración del tiempo de ejecución actualizada elimina un tipo de proceso del servidor, los procesos activos del servidor de ese tipo continúan ejecutándose hasta que terminan. La instancia no reemplaza esos procesos del servidor.
   - Si la configuración del tiempo de ejecución actualizada reduce el número de procesos simultáneos para un tipo de proceso del servidor: los procesos del servidor sobrantes de ese tipo continúan ejecutándose hasta que terminan. La instancia no reemplaza esos procesos del servidor sobrantes.
   - Si la configuración del tiempo de ejecución actualizada añade un nuevo tipo de proceso del servidor o aumenta los procesos simultáneos para un tipo existente, la instancia inicia los nuevos procesos del servidor hasta el número máximo de procesos en Amazon GameLift. En este caso, la instancia lanza nuevos procesos del servidor cuando los procesos existentes finalizan.

Optimización de una flota para varios procesos

Para utilizar varios procesos en una flota, realice el siguiente procedimiento:

- Cree una compilación que contenga todos los archivos ejecutables del servidor de juegos que desee implementar en una flota y cargue la compilación en Amazon GameLift. Todos los servidores de juegos de una compilación deben ejecutarse en la misma plataforma y utilizar el SDK de Amazon GameLift Server.
- Cree una configuración de tiempo de ejecución con una o varias configuraciones del proceso del servidor y múltiples procesos simultáneos.
- Integre los clientes de juegos con la versión 2016-08-04 o posterior del SDK de AWS.

Para optimizar el rendimiento de la flota, le recomendamos que realice las siguientes acciones:

- Gestionar los escenarios de cierre de procesos del servidor para que Amazon GameLift pueda reciclar los procesos de forma eficaz. Por ejemplo:
  - Añadir un procedimiento de cierre al código del servidor de juegos que llame a la API del servidor `ProcessEnding()`.
  - Implementar la función de devolución de llamada `OnProcessTerminate()` en el código del servidor de juegos para administrar las solicitudes de finalización de Amazon GameLift.
- Asegurarse de que Amazon GameLift cierre y vuelva a lanzar los procesos del servidor que no estén en buen estado. Informar del estado a Amazon GameLift mediante la implementación de la función de devolución de llamada de `OnHealthCheck()` en el código del servidor de juegos. Amazon GameLift cierra automáticamente los procesos del servidor en mal estado durante tres informes consecutivos. Si no implementa `OnHealthCheck()`, Amazon GameLift presupone que un proceso de servidor está en buen estado a menos que el proceso no responda a una comunicación.

Elección del número de procesos por instancia

Al decidir el número de procesos simultáneos que se van a ejecutar en una instancia, tenga en cuenta los siguientes aspectos:

- Amazon GameLift limita cada instancia a un número máximo de procesos simultáneos. La suma de todos los procesos simultáneos de las configuraciones de procesos de los servidores de una flota no puede superar esa cuota.
- Para mantener niveles de rendimiento aceptables, el tipo de instancia de Amazon EC2 puede limitar el número de procesos que pueden ejecutarse de forma simultánea Pruebe diferentes configuraciones del juego para encontrar el número correcto de procesos del tipo de instancia preferido.
- Amazon GameLift no ejecuta más procesos simultáneos que el número total configurado. Esto significa que la transición de la configuración del tiempo de ejecución anterior a la nueva configuración podría producirse de forma gradual.

Usa instancias puntuales con Amazon GameLift

Al configurar su flota de EC2 GameLift gestionada por Amazon, puede utilizar instancias puntuales, instancias bajo demanda o una combinación de ellas. Obtenga más información sobre cómo Amazon

GameLift utiliza las instancias puntuales enInstancias bajo demanda frente a instancias de spot. Para utilizar las flotas de spot, la integración del juego requiere los ajustes que se indican en esta página.

¿Lo estás utilizando FlexMatch para hacer matchmaking? Puede agregar flotas de spot a sus colas de sesiones de juego existentes para el emparejamiento.

1. Diseñe su cola de sesiones de juego para las instancias de spot.

   La administración de la ubicación de la sesión de juego con una cola es una práctica recomendada y es obligatoria al utilizar instancias de spot. Para diseñar la cola, tenga en cuenta lo siguiente:

   - Ubicaciones: para lograr la mejor experiencia de jugador, elija ubicaciones geográficamente cercanas a los jugadores.
   - Tipos de instancias: tenga en cuenta los requisitos de hardware de los servidores de juegos y la disponibilidad de las instancias en las ubicaciones que elija.

   Para probar una cola que optimice la disponibilidad y la resiliencia de spot, consulte Tutorial: Configuración de una cola de sesiones de juego para instancias de spot.
2. Cree las flotas para la cola optimizada para instancias de spot.

   En función del diseño de las colas, cree flotas para implementar sus servidores de juegos en las ubicaciones y tipos de instancias que desee. Consulte Creación de una flota administrada por Amazon GameLift para obtener ayuda sobre la creación y la configuración de flotas.
3. Cree una cola de sesión de juego.

   Añada los destinos de la flota, configure el proceso de ubicación de las sesiones de juego y defina las prioridades de ubicación. Consulte Creación de una cola de sesión de juego para obtener ayuda sobre la creación y la configuración de la nueva cola.
4. Actualice el servicio de cliente del juego para utilizar la cola.

   Cuando el cliente del juego utilice una cola para solicitar recursos, la cola evitará los recursos con una alta probabilidad de interrupción y seleccionará la ubicación que se ajuste a sus prioridades definidas. Para contribuir a la implementación de ubicaciones de sesión de juego en su cliente de juego, consulte Creación de sesiones de juego.
5. Actualice el servidor de juegos para gestionar una interrupción de instancias de spot.

AWS puede interrumpir las instancias puntuales con una notificación de 2 minutos cuando necesite recuperar su capacidad. Configure el servidor de juegos para gestionar las interrupciones y minimizar el impacto en los jugadores.

Antes de AWS recuperar una instancia puntual, envía una notificación de finalización. Amazon envía GameLift la notificación a todos los procesos del servidor afectados mediante la función de devolución de llamada GameLift del SDK de Amazon Server. `onProcessTerminate()` Implemente esta devolución de llamada para finalizar la sesión de juego o mover la sesión de juego y los jugadores a una nueva instancia. Consulte Respuesta a una notificación de cierre del proceso del servidor para obtener ayuda sobre la implementación `onProcessTerminate()`.

> **Note**
>
> AWS hace todo lo posible por enviar la notificación antes de recuperar una instancia, pero es posible que AWS recupere la instancia puntual antes de que llegue la advertencia. Prepare su servidor de juegos para gestionar interrupciones imprevistas.

6. Evalúe el rendimiento de las colas y las flotas de spot.

   Consulta GameLift las estadísticas de Amazon en la GameLift consola de Amazon o en Amazon CloudWatch para revisar el rendimiento. Para obtener más información sobre GameLift las métricas de Amazon, consultaSupervisión de Amazon GameLift con Amazon CloudWatch. Entre las métricas principales se incluyen:

   - Tasa de interrupción: utilice las métricas `InstanceInterruptions` y `GameSessionInterruptions` para realizar un seguimiento del número y frecuencia de las interrupciones relacionadas con spot para instancias y sesiones de juego. Las sesiones de juego que se reclaman por AWS tienen un estado `TERMINATED` y un motivo de estado de`INTERRUPTED`.
   - Efectividad de las colas: realice un seguimiento de las tasas de éxito en las ubicaciones, el tiempo medio de espera y la profundidad de las colas para comprobar que las flotas de spot no afectan al rendimiento de las colas.
   - Uso de la flota: supervise los datos sobre las instancias, las sesiones de juego y las sesiones de los jugadores. El uso de las flotas bajo demanda puede ser un indicador de que las colas no están utilizando ubicaciones en las flotas de spot para evitar interrupciones.

Creación de una nueva flota de Amazon GameLift

Cree una nueva flota e implemente su propia compilación de servidor de juegos o Realtime Servers como alojamiento. Puede implementar cualquier compilación de juego o recurso de script cargado en Amazon GameLift.

Temas

Cómo funciona la creación de flotas en Amazon GameLift

Al crear una flota nueva, Amazon GameLift inicia un flujo de trabajo que crea una flota con una instancia de Amazon Elastic Compute Cloud (Amazon EC2) en cada ubicación de la flota. A medida que Amazon GameLift completa cada paso del flujo de trabajo, la flota emite eventos y Amazon GameLift actualiza el estado de la flota. Puede realizar un seguimiento de todos los eventos mediante la consola de Amazon GameLift o llamando a la operación de la API de Amazon GameLift FleetCapacity. También puede realizar un seguimiento del estado de ubicaciones individuales mediante DescribeFleetLocationAttributes.

Flujo de trabajo de creación de la flota de EC2:

- Amazon GameLift crea un recurso de flota en la región de origen de la flota y en cada ubicación remota definida en la flota.
- Amazon GameLift establece la capacidad deseada en una instancia.
- Amazon GameLift establece el estado de la flota y la ubicación en Nuevo.
- Amazon GameLift comienza a escribir los eventos en el registro de eventos de la flota.
- Amazon GameLift asigna los recursos informáticos solicitados a una nueva instancia en cada ubicación de la flota.
- Amazon GameLift descarga los archivos del servidor de juegos en cada instancia y establece el estado de la flota en Descargando.
- Amazon GameLift valida los archivos del servidor de juegos descargados en cada instancia para verificar que no se ha producido ningún error durante la descarga. Amazon GameLift establece el estado de la flota en Validando.

- Amazon GameLift crea el servidor de juegos en cada instancia y establece el estado de la flota en Creando.
- Amazon GameLift se inicia al lanzar procesos de servidor en cada instancia, según las instrucciones de la configuración de tiempo de ejecución de la flota. Si ha configurado la flota para ejecutar varios procesos de servidor simultáneos por instancia, Amazon GameLift escalona el inicio del proceso unos segundos. A medida que se conecta cada proceso, informa a Amazon GameLift de que está listo. Amazon GameLift establece el estado de la flota en Activando.
- Amazon GameLift establece los estados de la flota y de la ubicación en Activo a medida que el servidor procesa la preparación del informe.

Amazon GameLift Anywhere fleet creation

- Amazon GameLift crea un recurso de flota. Para la región de origen de la flota y para cada ubicación personalizada definida en la flota, Amazon GameLift establece el estado de la flota y de la ubicación en Nueva.
- Amazon GameLift comienza a escribir los eventos en el registro de eventos de la flota.
- Cuando un proceso de servidor de una flota informa a Amazon GameLift de que está listo, Amazon GameLift establece el estado de la flota y de la ubicación en Activo. A medida que los procesos del servidor en otras ubicaciones de la flota informan de que están listos, Amazon GameLift establece el estado de cada ubicación de la flota en Activo.

Para obtener ayuda sobre problemas de creación de flotas, consulte Solución de problemas con la flota de Amazon GameLift.

Creación de una flota administrada por Amazon GameLift

Utilice la consola de Amazon GameLift o AWS Command Line Interface (AWS CLI) para crear una flota administrada.

Después de crear una nueva flota de EC2 administrada, el estado de la flota pasa a través de varias etapas a medida que Amazon GameLift implementa la flota e inicia los servidores de juegos. Una vez que la flota alcance el estado `ACTIVE`, está lista para alojar las sesiones de juego. Para obtener ayuda sobre problemas de creación de flotas, consulte Solución de problemas con la flota de Amazon GameLift.

Console

Para crear una flota de EC2 administrada, realice el siguiente procedimiento:

1. En la consola de Amazon GameLift, en el panel de navegación, elija Flotas.
2. En la página Flotas, elija Crear una flota.
3. Elija EC2 administrado.
4. En la página Detalles de la flota, haga lo siguiente:

   a. En Nombre, escriba un nombre para la flota. Le recomendamos que incluya el tipo de flota (spot o bajo demanda) en los nombres de la flota. Esto hará que sea mucho más sencillo identificar el tipo de flota al ver una lista de ellas.

   b. En Descripción, proporcione una breve descripción de la flota.

   c. En Tipo binario, seleccione Compilar o Script para definir el tipo de servidor de juegos que Amazon GameLift implementará en esa flota.

   d. Seleccione un script o una compilación de la lista desplegable de scripts o compilaciones cargados.

5. En Detalles adicionales para lo siguiente (opcional):

   a. En el caso del rol de instancia, especifique un rol de IAM que autorice a las aplicaciones de la compilación del juego a acceder a otros recursos de AWS de su cuenta. Para obtener más información, consulte Comunicación con otros recursos de AWS de sus flotas. Para crear una flota con un rol de instancia, la cuenta debe tener el permiso `PassRole` de IAM. Para obtener más información, consulte Ejemplos de permisos de IAM para Amazon GameLift.

   Si quiere autorizar aplicaciones no ejecutables de servidor, como un agente de CloudWatch, habilite la opción de credenciales compartidas.

   Esta configuración no se puede actualizar después de su creación.

   b. Para generar la certificación, elija que Amazon GameLift genere un certificado TLS para la flota. Puede utilizar un certificado TLS de flota para que el cliente de juego autentique un servidor de juegos cuando se conecte y cifre todas las comunicaciones cliente/servidor. Para cada instancia de una flota habilitada con TLS, Amazon GameLift también crea una nueva entrada de DNS con el certificado. Utilice estos recursos para configurar la autenticación y el cifrado para el juego.

c. En Grupo de métricas, introduzca el nombre de un grupo de métricas de flota nuevo o existente. Puede agregar las métricas de varias flotas añadiéndolas al mismo grupo de métricas.

   No puede actualizar el grupo de métricas después de crear la flota.

6. Elija Siguiente.
7. En la página Seleccionar ubicaciones, seleccione una o más ubicaciones remotas adicionales en las que implementar las instancias. La región de origen se selecciona automáticamente en función de la región desde la que accede a la consola. Si selecciona ubicaciones adicionales, las instancias de flota también se implementarán en estas ubicaciones.

   > **Important**
   >
   > Para utilizar las regiones que no estén habilitadas de forma predeterminada, actívelas en su Cuenta de AWS.
   >
   > - Las flotas con regiones que no estén habilitadas, y que haya creado antes del 28 de febrero de 2022, no se verán afectadas.
   > - Para crear nuevas flotas con varias ubicaciones o actualizar las existentes, habilite primero las regiones que desee utilizar.
   >
   > Para obtener más información sobre las regiones que no están habilitadas de forma predeterminada y cómo habilitarlas, consulte Administración de Regiones de AWS en la Referencia general de AWS.

8. Elija Siguiente.
9. En la página Definir detalles de la instancia, elija

   a. Instancias bajo demanda o Instancias de spot para esta flota. Para obtener más información sobre los tipos de flotas, consulte Instancias bajo demanda frente a instancias de spot.

   b. En el menú Arquitectura de filtros, elija x64 o ARM.

> **Note**
>
> Las instancias de ARM para Graviton requieren un servidor Amazon GameLift compilado en el sistema operativo Linux. Se requiere el SDK de servidor 5.1.1 o posterior para C++ y C#. Se requiere el SDK de servidor 5.1.1 o posterior para continuar. Estas instancias no admiten de forma inmediata la instalación de Mono en Amazon Linux 2023 (AL2023) o Amazon Linux 2 (AL2).

Para obtener información sobre las arquitecturas de ARM de Amazon EC2, consulte Procesador AWS Graviton y Tipos de instancia de Amazon EC2.

Para obtener información sobre los tipos de instancias compatibles con Amazon GameLift, consulte los valores de `EC2InstanceType` en los parámetros de solicitud de CreateFleet().

10. Seleccione un tipo de instancia de Amazon EC2 de la lista. Para obtener más información sobre cómo elegir un tipo de instancia, consulte Tipos de instancias. No se puede cambiar el tipo de instancia después de crear la flota.
11. Elija Siguiente.
12. En la página Configurar tiempo de ejecución, en Configuración del tiempo de ejecución, realice el siguiente procedimiento:

    a. En Ruta de lanzamiento, escriba la ruta al archivo ejecutable del juego en la compilación o script. En instancias Windows, los servidores de juegos se compilan en la ruta `C:\game`. En las instancias de Linux, los servidores de juegos están diseñados para `/local/game`. Ejemplos: **`C:\game\MyGame\server.exe`**, **`/local/game/MyGame/server.exe`** o **`MyRealtimeLaunchScript.js`**.

    b. En Parámetros de lanzamiento (opcional), introduzca la información para pasarla al archivo ejecutable del juego como un conjunto de parámetros de línea de comandos. Ejemplo: **`+sv_port 33435 +start_lobby`**.

    c. En el caso de los procesos simultáneos, seleccione el número de procesos del servidor que se ejecutarán simultáneamente en cada instancia de la flota. Revise los límites de Amazon GameLift en cuanto al número de procesos de servidor simultáneos.

Las restricciones a los procesos del servidor simultáneos por instancia se aplican a todos los procesos simultáneos de todas las configuraciones. Si configura la flota de manera que se supere el límite, la flota no podrá activarse.

13. En la sección Activación de la sesión de juego, indique los límites para activar nuevas sesiones de juego en las instancias de esta flota:

    a. En Número máximo de activaciones de sesiones de juego simultáneas, especifique el número de sesiones de juego en una instancia que se puedan activar al mismo tiempo. Este límite es útil cuando el lanzamiento de varias sesiones de juego nuevas puede afectar al desempeño de otras sesiones de juegos que se ejecutan en la instancia.

    b. En Tiempo de espera de la nueva activación, especifique cuánto tiempo debe esperar para que se active una sesión. Si la sesión de juego no pasa al estado `ACTIVE` anterior al tiempo de espera, Amazon GameLift finaliza la activación de la sesión de juego.

14. En Configuración del puerto EC2, realice el siguiente procedimiento (opcional):

    a. Haga clic en Agregar configuración de puertos para definir los permisos de acceso del tráfico entrante que conecta con el proceso del servidor implementado en la flota.

    b. En Tipo, elija TCP personalizado o UDP personalizado.

    c. En Rango de puertos, especifique un rango de números de puerto que permitan las conexiones entrantes. Un rango de puertos debe utilizar el formato `nnnnn[-nnnnn]` con valores entre 1026 y 60 000. Ejemplo: **`1500`** o **`1500-20000`**.

    d. En Rango de direcciones IP, especifique un rango de direcciones IP. Utilice la notación CIDR. Ejemplo: **`0.0.0.0/0`** (este ejemplo otorga acceso a cualquiera que intente conectarse).

15. En Configuración de los recursos de la sesión de juego, realice el siguiente procedimiento (opcional):

    a. En Política de protección para el escalado de juegos, active o desactive la protección escalable. Amazon GameLift no cerrará las instancias con protección durante un evento de reducción vertical si alojan una sesión de juego activa.

    b. En Límite de creación de recursos, especifique un número máximo de sesiones de juego que un jugador puede crear durante el periodo de la política.

16. Elija Siguiente.

17. Especifique los pares de clave y valor para añadir etiquetas a la compilación (opcional). Seleccione Siguiente para continuar con la revisión de la creación de la flota.

18. Seleccione Create (Crear). Amazon GameLift asigna un ID a la flota nueva e inicia el proceso de activación de la misma. Puede hacer un seguimiento del estado de la flota nueva en la página Flotas.

Puede actualizar los metadatos de la flota y la configuración en cualquier momento, independientemente del estado de la flota. Para obtener más información, consulte Administración de las flotas de Amazon GameLift. Puede actualizar la capacidad de la flota después de que la flota haya alcanzado el estado ACTIVO. Para obtener más información, consulte Escalación de la capacidad de alojamiento de Amazon GameLift. También puede añadir ubicaciones remotas o eliminarlas.

AWS CLI

Para crear una flota con la AWS CLI, abra una ventana de línea de comandos y utilice el comando `create-fleet`. Para obtener más información sobre el comando `create-fleet`, consulte `create-fleet` en la Referencia de comandos de la AWS CLI.

La solicitud `create-fleet` de ejemplo mostrada a continuación crea una nueva flota con las características siguientes:

- La flota utilizará instancias bajo demanda c5.large con el sistema operativo necesario para la compilación del juego seleccionada.
- Implementará la compilación del servidor de juegos especificada, que debe estar en estado Listo en las siguientes ubicaciones.
  - us-west-2 (región de origen)
  - sa-east-1 (ubicación remota)
- La generación de certificados TLS está habilitada.
- Cada instancia de la flota ejecutará diez procesos idénticos del servidor de juegos de forma simultánea, permitiendo que cada instancia aloje hasta diez sesiones de juego simultáneamente.
- En cada instancia, Amazon GameLift permitirá la activación de dos nuevas sesiones de juego al mismo tiempo. Asimismo, terminará cualquier sesión de juego de activación si no está lista para alojar jugadores en un plazo de 300 segundos.
- Todas las sesiones de juego alojadas en instancias en esta flota tienen la protección de sesión de juego activada.
- Los jugadores individuales pueden crear tres nuevas sesiones de juego en un periodo de 15 minutos.

- Cada sesión de juego alojada en esa flota tendrá un punto de conexión dentro de los rangos especificados de dirección IP y puerto.
- Amazon GameLift añadirá las métricas de esa flota al grupo de métrica de `EMEAfleets`, que (en este ejemplo) combina las métricas de todas las flotas en las regiones 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 solicitud de creación de la flota se realiza correctamente, Amazon GameLift devuelve un conjunto de atributos de la flota que incluye los valores de configuración solicitados y un ID de la flota nuevo. A continuación, Amazon GameLift inicia el proceso de activación de la flota y establece el estado de la flota y la ubicación en Nuevo. Puede realizar un seguimiento del estado de la flota y ver más información sobre la flota con estos comandos de la CLI:

- describe-fleet-events
- describe-fleet-attributes
- describe-fleet-capacity
- describe-fleet-port-settings
- describe-fleet-utilization
- describe-runtime-configuration

- describa los atributos de ubicación de la flota
- describe-fleet-location-capacity
- describe-fleet-location-utilization

Puede cambiar la capacidad de la flota y otras opciones de configuración según sea necesario mediante estos comandos:

- update-fleet-attributes
- update-fleet-capacity
- update-fleet-port-settings
- update-runtime-configuration
- create-fleet-locations
- delete-fleet-locations

Crea una GameLift Anywhere flota de Amazon

Usa Amazon GameLift para integrar el hardware de tu entorno en tu alojamiento de GameLift juegos de Amazon. Amazon GameLift Anywhere registra tu hardware con Amazon GameLift en una Anywhere flota. Puede integrar Anywhere y las flotas de EC2 administradas en las colas de sesión de juego y emparejador para gestionar la ubicación del juego y del emparejamiento.

Para obtener más información sobre cómo probar tus servidores de juegos con Amazon GameLift Anywhere, consultaRealización de una prueba de integración con flotas de Amazon GameLift Anywhere.

Para empezar, usa la Soporte de desarrollo con Amazon GameLift versión 5 o superior y revisa los siguientes conceptos para usar una GameLift Anywhere flota de Amazon.

Ubicaciones personalizadas

GameLift AnywhereLas flotas de Amazon utilizan ubicaciones personalizadas para representar las ubicaciones físicas de su infraestructura.

Registro de dispositivos

Para que una GameLift Anywhere flota de Amazon se comunique con tus recursos informáticos, primero registra tu dispositivo. Puedes completar el registro del dispositivo desde el Amazon

GameLift AWS SDK mediante la `RegisterCompute`operación. Esta operación utiliza la dirección IP del dispositivo para asociarlo a una ubicación de flota y comunicarse con Amazon GameLift.

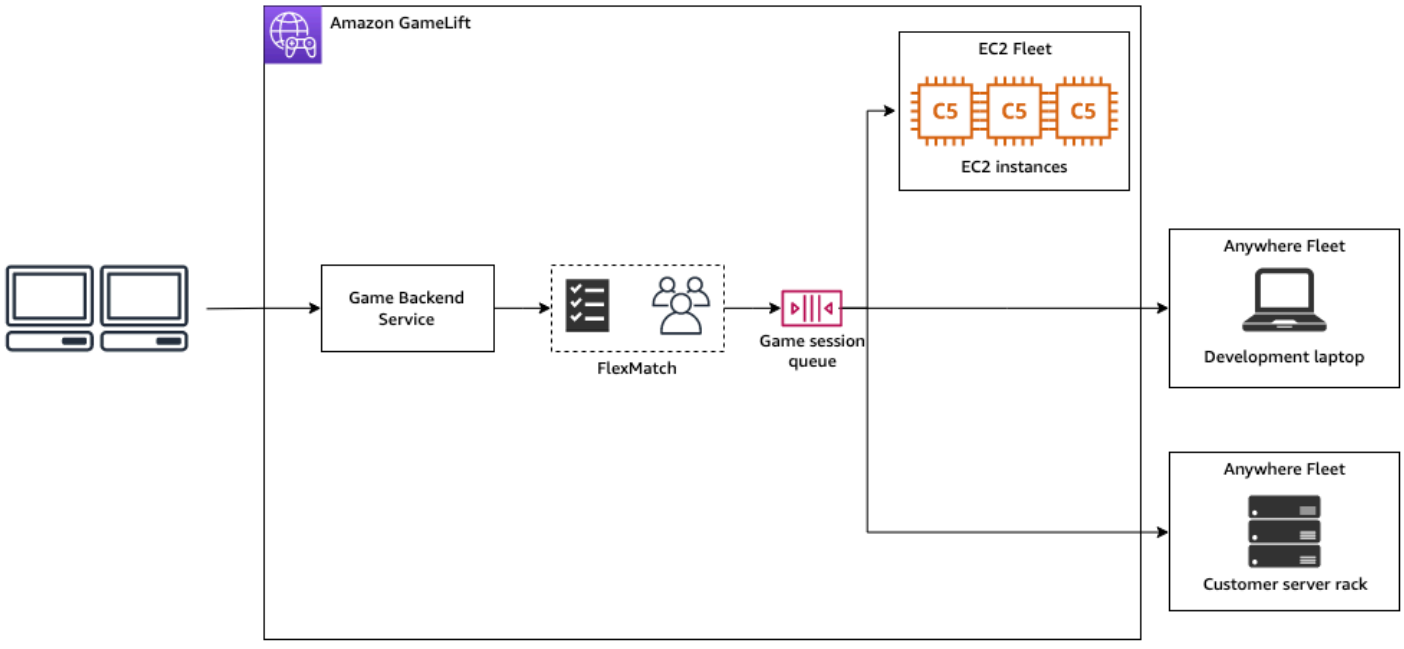
Tokens de autenticación

Cuando inicializas un servidor de juegos en tu ordenador, el SDK de Amazon GameLift Server utiliza un token de autenticación para autenticar tu servidor de juegos en Amazon. GameLift Puede volver a utilizar el mismo token de autenticación para todos los servidores de juegos en el mismo recurso informático hasta la fecha de caducidad del token de autenticación. Para recuperar el token de autenticación, ejecuta el comando (). `get-compute-auth-token` AWS Command Line Interface AWS CLI Pase el token a cada servidor de juegos según sea necesario.

Sesiones de juego

Cada sesión de juego de un recurso informático utiliza el mismo token de autenticación que se creó al registrar el recurso informático en una ubicación de flota.

En el siguiente diagrama se muestra una cola de sesiones de juego que utiliza el FlexMatch matchmaking y varias flotas. Las flotas incluyen una flota de EC2 con instancias C5, una flota deAnywhere con un portátil de desarrollo y una flota de Anywhere con un bastidor de servidores alojado por el cliente.



Temas

- Creación de una ubicación personalizada

- Creación de una flota
- Registro del recurso informático
- Ejecución de un proceso de servidor
- Creación de sesiones de juego
- Migración a un EC2 administrado

Creación de una ubicación personalizada

Para empezar a alojar juegos en sus recursos informáticos, cree una ubicación personalizada que describa dónde se encuentra el recurso informático.

Console

Para crear una ubicación personalizada, realice el siguiente procedimiento:

1. Abre la GameLift consola de Amazon.
2. En el panel de navegación, en Alojamiento, elija Ubicaciones.
3. En la página Locations (Ubicaciones), seleccione Create location (Crear ubicación).
4. En el cuadro de diálogo Crear ubicación, realice lo siguiente:
   a. Especifique el nombre de una ubicación. Esto etiqueta la ubicación del hardware que Amazon GameLift utiliza para ejecutar tus juegos en Anywhere flotas. Amazon GameLift añade custom - al nombre de tu ubicación personalizada.
   b. Añada etiquetas como pares clave-valor a su ubicación personalizada (opcional). Para cada etiqueta que desee añadir, elija Añadir etiqueta nueva.
   c. Seleccione Crear.

AWS CLI

Cree una ubicación personalizada mediante el comando `create-location`. `location-name`Etiqueta la ubicación del hardware que Amazon GameLift utiliza para ejecutar tus juegos en Anywhere las flotas. Al crear la ubicación personalizada, el nombre de la ubicación deberá empezar por `custom-`.

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

Salida

```
{
    "Location": {
        "LocationName": "custom-location-1",
        "LocationArn": "arn:aws:gamelift:us-east-1:111122223333:location/custom-
location-1"
    }
}
```

Creación de una flota

Usa la GameLift consola Amazon o la AWS CLI para crear una Anywhere flota.

Después de crear una nueva flota de Anywhere, el estado de la flota pasará de `NEW` a `ACTIVE`. Una vez alcance el estado `ACTIVE`, la flota estará lista para alojar las sesiones de juego. Para obtener ayuda sobre problemas de creación de flotas, consulte Solución de problemas con la flota de Amazon GameLift.

Console

Para crear una flota de Anywhere, realice el siguiente procedimiento:

1. Abre la GameLift consola de Amazon.
2. En el panel de navegación, en Alojamiento, elija Flotas.
3. En la página Flotas, elija Crear una flota.
4. En el paso Tipo informático, elija y Anywhere, a continuación, elija Siguiente.
5. En el paso Detalles de la flota, defina los detalles y, a continuación, elija Siguiente.
6. En el paso Ubicaciones personalizadas, seleccione la ubicación personalizada creada y, a continuación, elija Siguiente. Amazon selecciona GameLift automáticamente el hogar Región de AWS como la región en la que vas a crear la flota. Puede usar la región de origen para acceder a sus recursos y usarlos.
7. Complete el resto de los pasos de creación de la flota y, a continuación, seleccione Enviar para crear su flota de Anywhere.

AWS CLI

Crea una flota de Anywhere mediante el comando `create-fleet`. Incluya su ubicación personalizada en `locations`. Amazon GameLift crea la flota en tu región de origen y en las ubicaciones personalizadas que proporciones. En el siguiente ejemplo, reemplace *`FleetName`* y *`custom-location-1`* por su propia información. La variable `custom-location-1` es el nombre de la ubicación creada en el paso Creación de una ubicación personalizada.

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

Ejemplo de resultado

```
{
    "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"
    }
}
```

Registro del recurso informático

Para registrar el recurso informático en la flota creada, utilice el comando `register-compute`. Reemplace el *`fleet-id`* por el `fleet-id` devuelto en el paso anterior o por el ARN de flota que se encuentra en la página de detalles de su flota en la consola. Sustituya el *`compute-name`* y la *`ip-address`* por la dirección IP de su recurso informático.

> **Note**
>
> Te recomendamos ejecutar ambos `get-compute-auth-token` comandos desde un script o administrador de procesos independiente del servidor del juego. `register-compute`

```
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
```

Ejemplo de resultado

```
{
    "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"
    }
}
```

Ejecución de un proceso de servidor

1. Podrá obtener el token de autenticación para su recurso informático a partir de la flota creada.

   El servidor de juegos usa el token de autenticación para autenticarse en Amazon GameLift. Cada token de autenticación tiene una fecha de caducidad. Para seguir utilizando el recurso informático para alojar el servidor de juegos, recupere un nuevo token de autenticación antes de que caduque.

> **Note**
>
> Amazon GameLift recomienda llamar a los `get-compute-auth-token` comandos `register-compute` y desde un script o administrador de procesos independiente del servidor de juegos.

En el siguiente ejemplo, reemplace el `fleet-id` por el ARN o el ID de la flota de la flota creada en los pasos anteriores. Sustituya el `compute-name` por el nombre del recurso informático creado con el comando `register-compute` en un paso anterior.

```
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
```

Ejemplo de salida:

```
{
    "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. Ejecute una instancia del archivo ejecutable del servidor de juegos.

   Para ejecutar el servidor de juegos, inicialícelo llamando a `InitSDK()` y pasándole los parámetros del servidor. Para obtener más información, consulte ServerParameters.

   Entrada del SDK del servidor:

```
//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. Cuando el proceso del servidor esté listo para albergar una sesión de juego, llama `ProcessReady()` desde tu servidor de juegos a Amazon GameLift. Para obtener más información sobre los parámetros de procesamiento, consulte 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);
```

Creación de sesiones de juego

1. Añada lógica al servidor de juegos para que el proceso del servidor responda al mensaje `onStartGameSession()` con `ActivateGameSession()`. Esta operación no tiene parámetros, pero envía un acuse de recibo a Amazon de GameLift que tu servidor ha recibido y aceptado el mensaje de creación de sesión de juego.

```
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. Desde el servicio de backend del cliente del juego, inicie la sesión de juego con el comando `start-matchmaking`, `start-game-session-placement` o `create-game-session`.

```
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
```

Ejemplo de salida:

```
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 envía un on`StartGameSession()` mensaje al proceso de servidor registrado. El mensaje contiene el objeto `GameSession` del paso anterior con las propiedades del juego, los datos de las sesiones de juego, los datos del emparejador y más información sobre la sesión de juego.

3. Cuando complete la sesión de juego, finalice el proceso del servidor de juegos.

Entrada del SDK del servidor:

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

4. Inicie otro proceso del servidor de juegos llamando a `ProcessReady(processParams)`.

Migración a un EC2 administrado

Cuando hayas desarrollado tu servidor de juegos y estés preparado para la producción, puedes hacer que Amazon GameLift gestione tu hardware. Para migrar a una flota de EC2 gestionada, sube tu compilación a Amazon GameLift y crea una flota de EC2 gestionada. Para obtener más información sobre cómo cargar su compilación y configurar una flota, consulte Carga de una compilación del servidor de juegos personalizada en Amazon GameLift y Creación de una flota administrada por Amazon GameLift.

Administración de las flotas de Amazon GameLift

Utilice la consola de Amazon GameLift o la CLI de AWS para actualizar la configuración de la flota, cambiar las ubicaciones remotas o eliminar una flota.

Actualización de la configuración de una flota

Puede actualizar los atributos de la flota, los ajustes de los puertos y las configuraciones de tiempo de ejecución mutables mediante la consola de Amazon GameLift o la CLI de AWS. Para cambiar los límites de escalado, consulte Escalado automático de la capacidad con Amazon GameLift.

Amazon GameLift console

1. En la consola de Amazon GameLift, en el panel de navegación, elija Flotas.
2. Elija la flota que desee actualizar. Una flota debe tener el estado `ACTIVE` para poder editarla.
3. En la página de detalles de la flota, en cualquiera de las siguientes secciones, seleccione Editar.
    - Configuración de la flota
        - Modifique los atributos de la flota; por ejemplo, Name (Nombre) y Description (Descripción).
        - Añada o elimine grupos de métricas, que Amazon CloudWatch utiliza para realizar un seguimiento de las métricas agregadas de Amazon GameLift de varias flotas.
        - Actualice la configuración de los límites de creación de recursos.
        - Active o desactive la protección de sesiones de juego.

- Configuración del tiempo de ejecución: puede cambiar cualquiera de los siguientes ajustes de sus configuraciones de tiempo de ejecución y añadir o eliminar configuraciones de tiempo de ejecución.
  - Cambie la ruta de lanzamiento del servidor de juegos.
  - Añada, elimine o cambie los parámetros de lanzamiento opcionales.
  - Cambie el número de procesos simultáneos que ejecuten los servidores de juegos.
- Activación de la sesión de juego: cambie la forma en que desea que se ejecuten y alojen los procesos del servidor mediante la actualización de los campos Número máximo de activaciones de sesiones de juego simultáneas y Tiempo de espera de la nueva activación.
- Configuración de puertos EC2: actualice las direcciones IP y los rangos de puertos que permiten el acceso entrante a la flota.

4. Para guardar los cambios, elija Confirmar.

AWS CLI

Utilice los siguientes comandos de la CLI de AWS para actualizar una flota:

- update-fleet-attributes
- update-fleet-port-settings
- update-runtime-configuration

Actualización de ubicaciones de la flota

Puede añadir o eliminar las ubicaciones remotas de una flota mediante la consola de Amazon GameLift o la CLI de AWS. No se puede cambiar la región de origen de una flota.

Amazon GameLift console

1. En la consola de Amazon GameLift, en el panel de navegación, elija Flotas.
2. Elija la flota que desee actualizar. Una flota debe tener el estado `ACTIVE` para poder editarla.
3. En la página Detalles de la flota, elija la pestaña Ubicaciones para ver las ubicaciones de la flota.

4. Para añadir nuevas ubicaciones remotas, elija Añadir y seleccione las ubicaciones en las que desee implementar las instancias. Esta lista no incluye las instancias en las que el tipo de instancia de la flota no esté disponible.
5. Con las nuevas ubicaciones seleccionadas, elija Añadir. Amazon GameLift añade las nuevas ubicaciones a la lista con el estado establecido en `NEW`. A continuación, Amazon GameLift comienza a realizar un aprovisionamiento de una instancia en cada ubicación añadida y a prepararla para alojar sesiones de juego.
6. Para eliminar las ubicaciones remotas existentes de la flota, utilice las casillas de verificación para seleccionar una o más ubicaciones de la lista.
7. Con una o más flotas seleccionadas, elija Eliminar. Las ubicaciones eliminadas permanecen en la lista con el estado establecido en `DELETING`. Amazon GameLift inicia entonces el proceso de finalización de la actividad en la ubicación eliminada. Si hay instancias activas que alojan sesiones de juego, Amazon GameLift utiliza el proceso de finalización del servidor de juegos para finalizar correctamente las sesiones de juego y cerrar los servidores de juegos y las instancias.

AWS CLI

Utilice los siguientes comandos de la CLI de AWS para actualizar las ubicaciones de la flota:

- create-fleet-locations
- delete-fleet-locations

Eliminación de una flota

Puede eliminar una flota cuando ya no la necesite. Con la eliminación de una flota de forma permanente se borran todos los datos asociados con sesiones de juego y sesiones de jugador, así como los datos de las métricas recopilados. Como opción, puede conservar la flota, deshabilitar el escalado automático y escalar manualmente la flota a 0 instancias.

> **Note**
>
> Si la flota dispone de una conexión de emparejamiento de VPC, en primer lugar, llame a CreateVpcPeeringAuthorization para solicitar la autorización. Amazon GameLift borra la conexión de emparejamiento de VPC al eliminar la flota.

Puede utilizar la consola de Amazon GameLift o la herramienta de la CLI de AWS para eliminar una flota.

Amazon GameLift console

1. En la consola de Amazon GameLift, en el panel de navegación, elija Flotas.
2. Elija la flota que desee eliminar. Solo podrá eliminar las flotas que se encuentren en el estado `ACTIVE` o `ERROR`.
3. Elija Eliminar (Delete).
4. En el cuadro de diálogo Eliminar flota, introduzca **`delete`** para confirmar la eliminación.
5. Elija Eliminar (Delete).

AWS CLI

Utilice el siguiente comando de la CLI de AWS para eliminar una flota:

- delete-fleet

Añadir un alias a una GameLift flota de Amazon

Un GameLift alias de Amazon se utiliza para abstraer la designación de una flota. Las designaciones de flota indican a Amazon GameLift dónde buscar los recursos disponibles al crear nuevas sesiones de juego para los jugadores. Utilice alias en lugar de ID de la flota específicos para cambiar sin problemas el tráfico del jugador de una flota a otra cambiando la ubicación de destino del alias.

Existen dos tipos de estrategias de enrutamiento para alias:

- Simple: redirige el tráfico de jugadores a un ID de la flota especificado. Puede actualizar el ID de flota de un alias en cualquier momento.
- Terminal: transmite un mensaje al cliente. Por ejemplo, puedes dirigir a los jugadores que utilizan un out-of-date cliente a una ubicación en la que puedan obtener una mejora.

Las flotas tienen una vida útil finita y hay varias razones por las que cambiar de flotas durante un juego. No puede actualizar la versión del servidor de juegos de una flota ni cambiar determinados atributos de recursos informáticos en una flota existente. En su lugar, cree nuevas flotas con los

cambios y, a continuación, cambiar a los jugadores a las nuevas flotas. Con los alias, el cambio de flotas tiene un impacto mínimo sobre el juego y es invisible para los jugadores.

Los alias resultan útiles en los juegos que no utilizan colas. El cambio de flotas en una cola es simplemente cuestión de crear una nueva flota, añadirla a la cola y eliminar la flota antigua; ninguna de estas operaciones es visible para los jugadores. Por el contrario, los clientes de juegos que no utilizan colas deben especificar qué flota utilizar al comunicarse con el GameLift servicio de Amazon. Sin los alias, un cambio de flota requiere hacer actualizaciones en el código del juego y posiblemente en la distribución de clientes de juego actualizados a los jugadores.

Al actualizar el ID de la flota al que hace referencia un alias, hay un periodo de transición de hasta dos minutos en el que las sesiones de juego con el alias pueden terminar en la flota anterior.

Creación de un alias nuevo

Puede crear un alias mediante la GameLift consola de Amazon, tal y como se describe aquí, o con el comando create-alias de la AWS CLI.

1. En la GameLift consola de Amazon, en el panel de navegación, selecciona Aliases.
2. En la página Alias, elija Crear alias. Le recomendamos incluir el tipo de flota en los nombres de alias. Esto hará que sea mucho más sencillo identificar el tipo de flota al ver una lista de alias.
3. En la página Crear alias, en Detalles de alias, haga lo siguiente:
   a. En Nombre, introduzca un nombre de alias.
   b. Para obtener una descripción, especifique una descripción breve para su identificación.
   c. Elija el tipo de enrutamiento Simple o Terminal.
4. En Etiquetas, introduzca los pares Clave y Valor para añadir etiquetas al alias (opcional).
5. Seleccione Crear.

Edición de un alias

Puede editar un alias mediante la GameLift consola de Amazon o con el comando update-alias de la AWS CLI.

1. En la GameLift consola de Amazon, en el panel de navegación, selecciona Aliases.
2. En la página Alias, elija el alias que desee editar.
3. En la página de alias, seleccione Editar.

4. En la página Edit alias, puede editar los siguientes campos:

   - Nombre de alias: es el nombre fácil de recordar del alias.
   - Descripción: es una descripción breve del alias.
   - Tipo: es la estrategia de direccionamiento del tráfico de jugadores. Seleccione Simple para cambiar la flota asociada o seleccione Terminal para editar el mensaje de terminación.

5. Elija Guardar cambios.

Solución de problemas con la flota de Amazon GameLift

En este tema se proporcionan instrucciones sobre cómo solucionar problemas relacionados con la configuración de flotas para una solución de Amazon GameLift administrada. Para solucionar problemas adicionales, puede acceder de forma remota a una instancia de la flota una vez que la flota esté activa. Consulte Conéctese remotamente a las instancias de GameLift la flota de Amazon.

Problemas al crear una flota

Cuando se crea una flota, el servicio de Amazon GameLift inicia un flujo de trabajo que implementa una nueva instancia en cada una de las ubicaciones de la flota y la prepara para ejecutar los servidores de juegos. Para obtener una descripción detallada, consulte Cómo funciona la creación de flotas en Amazon GameLift. Una flota no puede albergar sesiones de juego ni jugadores hasta que alcance el estado Activo. En esta sección se analizan los problemas más comunes que impiden que las flotas se activen.

Descarga y validación

Durante esta fase, pueden producirse errores en la creación de la flota si hay problemas con los archivos de compilación extraídos, el script de instalación no se ejecuta o los archivos ejecutables designados en la configuración del tiempo de ejecución no están incluidos en los archivos de compilación. Amazon GameLift proporciona registros relacionados con cada uno de estos problemas.

Si los registros no revelan incidencias, es posible que el problema se deba a un error de servicio interno. En este caso, intente crear de nuevo la flota. Si el problema persiste, considere la posibilidad de volver a cargar la compilación del juego (en caso de que los archivos estén dañados). También puede ponerse en contacto con el equipo de soporte de Amazon GameLift o publicar una pregunta en el foro.

Building

En la mayoría de los casos, las incidencias que producen errores durante la fase de compilación se deben a problemas con los archivos de compilación del juego o con el script de instalación. Compruebe que los archivos de compilación del juego, tal y como se cargaron en Amazon GameLift, pueden instalarse en un equipo que ejecute el sistema operativo apropiado. Asegúrese de utilizar una instalación limpia del sistema operativo y no un entorno de desarrollo existente.

Activación

Los problemas más habituales de creación de flotas se producen durante la fase Activating (Activando). Durante esta fase, se comprueban una serie de elementos, como la viabilidad del servidor de juegos, la configuración del tiempo de ejecución y la capacidad del servidor de juegos de interactuar con el servicio de Amazon GameLift mediante el SDK del servidor. Entre los problemas habituales que surgen durante la activación de la flota se incluyen:

Los procesos del servidor no arrancan.

En primer lugar, compruebe que ha definido correctamente la ruta de lanzamiento y los parámetros de lanzamiento opcionales en la configuración del entorno en tiempo de ejecución de la flota. Puede ver la configuración del tiempo de ejecución actual de la flota mediante la página de detalles Flota, sección Detalles) o llamando al comando de la AWS CLI describe-runtime-configuration. Si la configuración del motor en tiempo de ejecución es correcta, compruebe si hay problemas con los archivos de compilación del juego o con el script de instalación.

Los procesos del servidor arrancan pero la flota no se activa.

Si los procesos del servidor arrancan y se ejecutan correctamente, pero la flota no pasa al estado Activo, es probable que la causa sea que el proceso del servidor no esté informando a Amazon GameLift de que está listo para alojar las sesiones de juego. Compruebe que el servidor de juegos llame correctamente a la acción de la API de servidor `ProcessReady()` (consulte Inicialización del proceso del servidor).

Error de solicitud de interconexión con la VPC.

Para las flotas que se crean con una interconexión de VPC (consulte Para configurar la interconexión de VPC con una nueva flota), la interconexión de VPC se realiza durante esta fase Activating (Activando). Si una interconexión de VPC devuelve un error por cualquier motivo, la flota nueva no podrá pasar al estado Activo. Puede comprobar si la solicitud de interconexión se realiza o no correctamente llamando a describe-vpc-peering-connections. Asegúrese de verificar que existe una autorización de interconexión de VPC válida (describe-vpc-peering-authorizations, ya que las autorizaciones solo son válidas durante 24 horas.

Problemas con los procesos del servidor

Los procesos del servidor arrancan pero dan error rápidamente o notifican que no están funcionando correctamente.

Aparte de los problemas con la compilación del juego, esto puede ocurrir al intentar ejecutar demasiados procesos del servidor simultáneamente en la instancia. La cantidad óptima de procesos simultáneos depende del tipo de instancia y de los requisitos de los recursos del servidor de juegos. Intente reducir el número de procesos simultáneos, que se define en la configuración del tiempo de ejecución de la flota, para ver si mejora el desempeño. Puede cambiar la configuración del tiempo de ejecución de una flota mediante la consola de Amazon GameLift (edite la configuración de asignación de capacidad de la flota) o llamando al comando de la AWS CLI update-runtime-configuration.

Problemas de eliminación de la flota

La flota no se puede terminar debido a un recuento máximo de instancias.

El mensaje de error indica que la flota que se está eliminando todavía tiene instancias activas, lo que no está permitido. En primer lugar, debe reducir una flota a cero instancias activas. Esto se realiza configurando manualmente el recuento de instancia de la flota deseado a "0" y, a continuación, esperar a que la reducción de escala surta efecto. Asegúrese de desactivar el escalado automático, que contrarrestará los ajustes manuales.

Las acciones de VPC no están autorizadas.

Este problema solo se aplica a las flotas para las que haya creado específicamente conexiones de emparejamiento de VPC (consulte Emparejamiento de VPC para Amazon GameLift). Este escenario se produce porque el proceso de eliminación de una flota también incluye la eliminación de la VPC de la flota y de cualquier conexión de emparejamiento de la VPC. Debe obtener una autorización primero llamando a la API del servicio de Amazon GameLift CreateVPCPeeringAuthorization() o mediante el comando de la CLI de AWS `create-vpc-peering-authorization`. Una vez que tenga la autorización, puede eliminar la flota.

Problemas con la flota de Realtime Servers

Sesiones de juego zombies: inician y ejecutan un juego, pero nunca terminan.

Es posible que observe estos problemas como cualquiera de las siguientes situaciones:

- Los servidores de Realtime de la flota no recogen las actualizaciones de script.
- La flota alcanza rápidamente la capacidad máxima y no se reduce cuando disminuye la actividad de los jugadores (como, por ejemplo, las solicitudes de sesiones de nuevo juego).

Esto es casi seguro resultado de no conseguir una llamada correcta en el script de Realtime. Aunque la flota pasa a estar activa y las sesiones de juego se inician, no hay ningún método para detenerlas. Como resultado, el servidor de Realtime que está ejecutando la sesión de juego nunca se libera para iniciar una nueva y solo pueden iniciarse las nuevas sesiones de juego cuando se activan nuevos servidores de Realtime. Además, las actualizaciones en el script de Realtime no afectan a las sesiones de juego que ya están ejecutándose.

Para evitar que ocurra esto, los scripts tienen que proporcionar un mecanismo para activar una llamada `processEnding`. Tal y como se muestra en la Ejemplo del script de Realtime Servers, una forma consiste en programar un tiempo de espera de sesión inactiva donde, si no se conecta ningún jugador durante un cierto tiempo, el script finalizará la sesión de juego actual.

Sin embargo, si se encuentra en esta situación, hay un par de soluciones para desbloquear los servidores de Realtime. El truco consiste en activar los procesos del servidor de Realtime (o las instancias subyacentes de la flota) para que se reinicien. En este caso, GameLift cierra automáticamente las sesiones de juego. Una vez que se liberan los servidores de Realtime, pueden iniciar nuevas sesiones de juego mediante la versión más reciente del script de Realtime.

Hay un par de métodos para lograrlo, en función de lo generalizado que esté el problema:

- Reducir toda la flota. Este es el método es el más sencillo, pero tiene un efecto amplio. Reduzca la flota a cero instancias, espere a que la flota se reduzca totalmente y, a continuación, vuelva a aumentarla. Esto eliminará todas las sesiones de juego existentes y permitirá iniciar de cero con el script de Realtime actualizado más recientemente.
- Acceda de forma remota a la instancia y reinicie el proceso. Se trata de una buena opción si tiene que corregir unos cuantos procesos. Si ya está conectado a la instancia, como, por ejemplo, a los registros de cola o depuración, este puede ser el método más rápido. Consulte Conéctese remotamente a las instancias de GameLift la flota de Amazon.

Si decide no incluir una forma de llamar a `processEnding` en el script de Realtime, hay varias situaciones complicadas que podrían producirse incluso cuando la flota se activa y se inician las sesiones de juego. En primer lugar, la ejecución de una sesión de juego no finaliza. Como resultado, el proceso del servidor que está ejecutando dicha sesión de juego no está libre nunca

para iniciar una nueva sesión de juego. En segundo lugar, el servidor de Realtime no recoge ninguna actualización de script.

Conéctese remotamente a las instancias de GameLift la flota de Amazon

Puede conectarse a cualquier instancia de sus flotas EC2 activas GameLift gestionadas por Amazon. Entre los motivos habituales para acceder a una instancia se incluyen los siguientes:

- Soluciona problemas relacionados con la integración del servidor de juegos
- Ajusta la configuración del tiempo de ejecución y otros ajustes específicos de la flota
- Obtén información sobre la actividad del servidor de juegos en tiempo real, como el seguimiento de los registros.
- Ejecuta herramientas de evaluación comparativa utilizando el tráfico real de jugadores.
- Investiga problemas específicos relacionados con una sesión de juego o un proceso del servidor.

Cuando te conectes a una instancia, ten en cuenta estos posibles problemas:

- Puede conectarse a instancias de flotas activas. Es posible que se pueda acceder a las flotas inactivas, las que se estén activando o se encuentren en un estado de error, durante un breve período de tiempo. Si necesita ayuda en relación con los problemas de activación de la flota, consulte Solución de problemas con la flota de Amazon GameLift.
- La conexión a una instancia activa no afecta a la actividad de alojamiento de la instancia. La instancia sigue iniciando y deteniendo los procesos del servidor en función de la configuración del tiempo de ejecución. Activa y aloja la sesión de juego. Es posible que se cierre en respuesta a un evento de reducción de escala u otro evento.
- Cualquier cambio que realices en los archivos o la configuración de la instancia podría afectar a las sesiones de juego activas de la instancia y a los jugadores conectados.

Las siguientes instrucciones describen cómo conectarse remotamente a una instancia mediante la interfaz de línea de AWS comandos (CLI). También puedes realizar llamadas programáticas mediante el AWS SDK, tal y como se documenta en la referencia de la API GameLift de Amazon Service.

Recopila datos de la instancia

Recopile la siguiente información:

- El ID de la instancia a la que quieres conectarte. Puede usar el ID de instancia o el ARN.
- La versión GameLift del SDK del servidor Amazon que se utiliza en la instancia. El SDK del servidor está integrado con la compilación del juego que se ejecuta en la instancia.

Para recuperar los datos de la instancia

En los siguientes pasos se supone que tiene un ID de flota de EC2 gestionado para la instancia a la que desea conectarse.

1. Obtenga el nombre del equipo.

   Llame a list-compute para la flota de EC2 gestionada para obtener una lista de todos los ordenadores activos de la flota. Para una flota de una sola ubicación, especifique el ID o el ARN de la flota. Para una flota con varias ubicaciones, especifique el ID o ARN de la flota y una ubicación. En el caso de una flota de EC2 gestionada, los cálculos son instancias de EC2 y la propiedad `ComputeName` devuelta es el ID de la instancia. Por ejemplo:

   Solicitud

   ```
   aws gamelift list-compute \
     --fleet-id  "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa" \
     --location ""sa-east-1"
   ```

   Respuesta

   ```
   {
     "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. Busque la versión del SDK del servidor.

   La versión del SDK del servidor es un atributo de un recurso de compilación.

   a. Llame describe-fleet-attributescon un identificador de flota para obtener el identificador de construcción y el ARN de la flota.

   b. Llama a describe-build con el ID de compilación o el ARN para obtener la versión del SDK del servidor de la compilación.

      Por ejemplo:

      Solicitud

```
aws gamelift describe-fleet-attributes /
  --fleet-ids  "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa"
```

Respuesta

```
{
  "FleetAttributes": [
    {
      "FleetId": "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa",
      "ComputeType": "EC2",
      "BuildId": "build-3333cccc-44dd-55ee-66ff-00001111aa22",
      . . .
    }
  ]
}
```

Solicitud

```
aws gamelift describe-build /
  --build-id "build-3333cccc-44dd-55ee-66ff-00001111aa22"
```

Respuesta

```
"Build": {
  "BuildId": "build-1111aaaa-22bb-33cc-44dd-5555eeee66ff",
  "Name": "My_Game_Server_Build_One",
  "OperatingSystem": "AMAZON_LINUX_2",
  "ServerSdkVersion": "5.1.1",
  . . .
}
```

Conectarse a una instancia (SDK de servidor 5)

Si la instancia a la que quieres conectarte ejecuta una compilación de juego con el SDK de servidor versión 5.x, sigue las instrucciones siguientes para conectarte a la instancia mediante Amazon EC2 Systems Manager (SSM). Puede acceder a las instancias remotas que se ejecuten en Windows o Linux.

1. Solicite credenciales de acceso para la instancia. Cuando tengas un nombre informático y un ID de flota para la instancia a la que quieres conectarte, llama. get-compute-access Si se ejecuta correctamente, Amazon GameLift devuelve un conjunto de credenciales temporales para acceder a la instancia. Por ejemplo:

   Solicitud

   ```
   aws gamelift get-compute-access \
   --compute-name i-11111111a222b333c \
   --fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa
   --region us-west-2
   ```

   Respuesta

   ```
   {
     "ComputeName": " i-11111111a222b333c ",
     "Credentials": {
       "AccessKeyId": " ASIAIOSFODNN7EXAMPLE ",
       "SecretAccessKey": " wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY ",
       "SessionToken": " AQoDYXdzEJr...<remainder of session token>"
     },
     "FleetArn": " arn:aws:gamelift:us-west-2::fleet/
   fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa ",
     "FleetId": " fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa "
   ```

```
}
```

2. Exporte las credenciales de acceso. Si lo desea, puede exportar las credenciales a variables de entorno y utilizarlas para configurar la AWS CLI para el usuario predeterminado. Para obtener más información, consulte Variables de entorno para configurar la AWS CLI en la Guía del AWS Command Line Interface usuario.

```
export AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of session token>
```

3. Conéctese a la instancia de flota. Inicie una sesión de SSM con la instancia a la que desee conectarse. Incluye la AWS región o la ubicación de la instancia. Para obtener más información, consulte Iniciar una sesión (AWSCLI) en la Guía del usuario de Amazon EC2 Systems Manager. Utilice las credenciales que adquirió en el paso 1. Por ejemplo:

```
aws ssm start-session \
--target i-11111111a222b333c \
--region us-west-2
```

Conectarse a una instancia (SDK de servidor 4.x o anterior)

Si la instancia a la que quieres conectarte ejecuta una compilación de juego con la versión 4 o anterior del SDK de servidor, sigue las instrucciones siguientes. Puedes conectarte a instancias que ejecuten Windows o Linux. Conéctese a una instancia de Windows mediante un cliente de protocolo de escritorio remoto (RDP). Conéctate a una instancia de Linux mediante un cliente SSH.

1. Solicite credenciales de acceso para la instancia. Cuando tengas un ID de instancia, usa el comando get-instance-accesspara solicitar las credenciales de acceso. Si se ejecuta correctamente, Amazon GameLift devuelve el sistema operativo de la instancia, la dirección IP y un conjunto de credenciales (nombre de usuario y clave secreta). El formato de las credenciales depende del sistema operativo de la instancia. Siga las siguientes instrucciones para recuperar las credenciales para RDP o SSH.

   - Para instancias Windows: para conectarse a una instancia Windows, RDP requiere un nombre de usuario y una contraseña. La solicitud `get-instance-access` devuelve estos valores en forma de cadenas simples, por lo que puede utilizar los valores devueltos tal cual. Credenciales de ejemplo:

```
"Credentials": {
    "Secret": "aA1bBB2cCCd3EEE",
    "UserName": "gl-user-remote"
}
```

- Para instancias Linux: para conectarse a una instancia Linux, SSH requiere un nombre de usuario y una clave privada. Amazon GameLift emite claves privadas de RSA y las devuelve como una cadena única, con el carácter de nueva línea (\n) que indica los saltos de línea. Para que la clave privada sea utilizable, siga estos pasos: (1) convierta la cadena en un `.pem` archivo y (2) establezca los permisos para el nuevo archivo. Credenciales de ejemplo devueltas:

```
"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"
}
```

Al usar la AWS CLI, puede generar automáticamente un `.pem` archivo al incluir los parámetros --query y --output en su `get-instance-access` solicitud.

Para establecer permisos en el nuevo archivo `.pem`, ejecute el siguiente comando:

```
$ chmod 400 MyPrivateKey.pem
```

2. Abra un puerto para la conexión remota. Puedes acceder a las instancias de las GameLift flotas de Amazon a través de cualquier puerto autorizado en la configuración de la flota. Puede ver la configuración de los puertos de la flota mediante el comando `describe-fleet-port-settings`.

   Recomendamos abrir los puertos para el acceso remoto solo cuando se necesiten y cerrarlos cuando haya terminado. No puedes actualizar la configuración de los puertos después de crear una flota, sino antes de que esté activa. Si te quedas atascado, vuelve a crear la flota con la configuración del puerto abierta.

   Utilice el comando `update-fleet-port-settings` para añadir una configuración de puerto para la conexión remota (como 22 para SSH o 3389 para RDP). Para el valor del rango de IP, especifique las direcciones IP de los dispositivos que desea utilizar para conectarse (convertidos al formato CIDR). Ejemplo:

   ```
   $ 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"
   ```

   En el siguiente ejemplo se abre el puerto 3389 en una flota de Windows.

   ```
   $ AWS gamelift update-fleet-port-settings
   --fleet-id  "fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa"
       --inbound-permission-authorizations
    "FromPort=3389,ToPort=3389,IpRange=54.186.139.221/32,Protocol=TCP"
   ```

3. Abra un cliente de conexión remoto. Utilice el escritorio remoto para Windows o SSH para instancias Linux. Conéctese a la instancia a través de la dirección IP, la configuración del puerto y las credenciales de acceso.

   Ejemplo de SSH:

```
ssh -i MyPrivateKey.pem gl-user-remote@192.0.2.0
```

Vea los archivos en instancias remotas

Al conectarse a una instancia de forma remota, dispone de acceso administrativo y de usuario pleno. Esto significa que también tendrá la capacidad de provocar errores y fallos al alojar juegos. Si la instancia aloja juegos con jugadores activos, corres el riesgo de bloquear las sesiones de juego y perder jugadores, o de interrumpir los procesos de cierre del juego y provocar errores en los datos y registros guardados de las partidas.

Busca estos recursos en una instancia de alojamiento:

- Archivos de compilación de juegos. Estos archivos son la versión del juego que subiste a Amazon GameLift. Incluyen uno o más ejecutables, activos y dependencias del servidor de juegos. Los archivos de compilación del juego se encuentran en un directorio raíz llamado: `game`
  - En Windows: `c:\game`
  - En Linux: `/local/game`
- Archivos de log de juegos. Busca los archivos de registro que genera tu servidor de juegos en el directorio `game` raíz, en la ruta de directorio que hayas designado.
- Recursos de GameLift alojamiento de Amazon. El directorio raíz `Whitewater` contiene los archivos que utiliza el GameLift servicio de Amazon para gestionar la actividad de alojamiento de juegos. No modifiques estos archivos por ningún motivo.
- Configuración de tiempos de ejecución. No accedas a la configuración del tiempo de ejecución para instancias individuales. Para realizar cambios en una propiedad de configuración de tiempo de ejecución, actualice la configuración de tiempo de ejecución de la flota (consulte la operación del AWS SDK UpdateRuntimeConfigurationo la AWS CLI update-runtime-configuration).
- Datos de la flota. Un archivo JSON contiene información sobre la flota a la que pertenece la instancia, para que la utilicen los procesos del servidor que se ejecutan en la instancia. El archivo JSON se encuentra en la siguiente ubicación:
  - En Windows: `C:\GameMetadata\gamelift-metadata.json`
  - En Linux: `/local/gamemetadata/gamelift-metadata.json`
- Certificados TLS. Si la instancia pertenece a una flota que tiene habilitada la generación de certificados TLS, busca los archivos de certificados, incluidos el certificado, la cadena de certificados, la clave privada y el certificado raíz, en la siguiente ubicación:

- En Windows: `c:\\GameMetadata\Certificates`
- En Linux: `/local/gamemetadata/certificates/`

Escalación de la capacidad de alojamiento de Amazon GameLift

La capacidad de alojamiento, medida en instancias, representa el número de sesiones de juego que Amazon GameLift puede alojar simultáneamente y el número de jugadores simultáneos que pueden alojar esas sesiones de juego. Una de las tareas más complicadas del alojamiento de juegos es escalar la capacidad para satisfacer la demanda de los jugadores sin malgastar dinero en recursos innecesarios. Para obtener más información, consulte Escalado de la capacidad de la flota.

La capacidad se ajusta a nivel de ubicación de la flota. Todas las flotas tienen al menos una ubicación: la región de AWS de origen de la flota. Al ver o escalar la capacidad, la información se muestra por ubicación, incluida la región de origen de la flota y cualquier otra ubicación remota.

Puede configurar manualmente el número de instancias que desee mantener, o puede configurar el escalado automático para ajustar la capacidad de forma dinámica a medida que la demanda de los jugadores cambia. Le recomendamos que comience activando la opción de escalado automático basado en objetivos. La función del escalado automático basado en objetivos es mantener suficientes recursos de alojamiento para dar cabida a los jugadores actuales y a algunos más con el fin de administrar los picos inesperados en la demanda de jugadores. Para la mayoría de los juegos, el escalado automático basado en objetivos ofrece una solución de escalado muy eficaz.

Los temas de esta sección proporcionan ayuda en detalle en relación con las siguientes tareas:

- Establecimiento de límites máximos y mínimos para el escalado de la capacidad
- Establecimiento manual de los niveles de capacidad
- Uso del escalado automático basado en objetivos
- Administración del escalado automático basado en reglas (característica avanzada)
- Desactivación temporal del escalado automático

La mayoría de las actividades de escalado de la flota se pueden realizar con la consola de Amazon GameLift. También puede utilizar un SDK de AWS o la AWS Command Line Interface (AWS CLI) con la API de servicio de Amazon GameLift.

Para administrar la capacidad de la flota en la consola, realice el siguiente procedimiento:

1. Abra la consola de Amazon GameLift.
2. En el panel de navegación, elija Alojamiento y Flotas.
3. En la página Flotas, elija el nombre de una flota activa para abrir la página de detalles de la flota.
4. Elija la pestaña Escalado. En esa pestaña, podrá realizar las siguientes acciones:
   - Consultar las métricas de escalado históricas de toda la flota.
   - Ver y actualizar la configuración de capacidad de cada ubicación de la flota, incluidos los límites de escalado y la configuración de la capacidad actual.
   - Actualizar el escalado automático basado en objetivos, consultar las políticas de escalado automático basado en reglas que se aplican a toda la flota y suspender la actividad de escalado automático en cada ubicación.

Temas

Establecimiento de los límites de capacidad de Amazon GameLift

Al escalar la capacidad de alojamiento de una ubicación de la flota de Amazon GameLift, ya sea de forma manual o con un escalado automático, tenga en cuenta los límites de escalado de la ubicación. Todas las ubicaciones de la flota tienen un límite mínimo y máximo que definen el rango permitido para la capacidad de la ubicación. De forma predeterminada, los límites de las ubicaciones de la flota establecen en un mínimo de 0 instancias y un máximo de 1. Para poder escalar la ubicación de una flota, ajuste los límites.

Si utiliza el escalado automático, el límite máximo permitirá a Amazon GameLift escalar verticalmente la ubicación de una flota para satisfacer la demanda de los jugadores, pero evita costos de alojamiento desmesurados, como durante un ataque DDOS. Configure una alarma de Amazon CloudWatch para que le informe cuando la capacidad se acerque al límite máximo con el fin de poder evaluar la situación y realizar ajustes manuales según sea necesario. (También puede crear una

alarma de facturación para controlar los costos de AWS). El límite mínimo resulta útil para mantener la disponibilidad del alojamiento, incluso cuando la demanda de jugadores es baja.

Puede establecer límites de capacidad para las ubicaciones de una flota en la consola de Amazon GameLift o mediante la AWS Command Line Interface (AWS CLI).

Para establecer límites de capacidad.

Console

1. Abra la consola de Amazon GameLift.
2. En el panel de navegación, elija Alojamiento y Flotas.
3. En la página Flotas, elija el nombre de una flota activa para abrir la página de detalles de la flota.
4. En la pestaña Escalado, en Capacidad de escalado, seleccione una ubicación de la flota y, a continuación, seleccione Editar.
5. En el cuadro de diálogo Editar capacidad de escalado, establezca los recuentos de instancias para Tamaño mínimo, Instancias deseadas y Tamaño máximo.
6. Elija Confirmar.

AWS CLI

1. Compruebe la configuración de la capacidad actual. En una ventana de línea de comandos, utilice el comando describe-fleet-location-capacity con el ID de la flota y la ubicación para los que quiera cambiar la capacidad. Este comando devuelve un objeto FleetCapacity que incluye la configuración de la capacidad actual de la ubicación. Determine si los nuevos límites de instancia se ajustarán a la configuración de instancias deseadas actual.

```
aws gamelift describe-fleet-location-capacity \
    --fleet-id <fleet identifier> \
    --location <location name>
```

2. Actualice la configuración de límites. En una ventana de línea de comandos, utilice el comando update-fleet-capacity con los siguientes parámetros. Puede ajustar tanto los límites de instancias como el recuento de instancias deseadas con el mismo comando.

```
--fleet-id <fleet identifier>
--location <location name>
```

```
--max-size <maximum capacity for scaling>
--min-size <minimum capacity for scaling>
--desired-instances <fleet capacity goal>
```

Ejemplo:

```
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 la solicitud se realiza correctamente, Amazon GameLift devuelve el ID de la flota. Si el valor nuevo `max-size` o el `min-size` valor entran en conflicto con la `desired-instances` configuración actual, Amazon GameLift devuelve un error.

Configuración manual de la capacidad de una flota de Amazon GameLift

Al crear una flota nueva, Amazon GameLift establece automáticamente las instancias deseadas en una instancia en cada ubicación de la flota. A continuación, Amazon GameLift implementa una nueva instancia en cada ubicación. Para cambiar la capacidad de la flota, puede añadir una política de escalado automático basada en objetivos o configurar manualmente la cantidad de instancias que desea para una ubicación. Para obtener más información, consulte Escalado de la capacidad de la flota.

Establecer la capacidad de una flota manualmente puede resultar útil cuando no necesita aplicar el escalado automático o cuando precisa mantener la capacidad en un nivel específico. La configuración manual de la capacidad solo funciona si no utiliza una política de escalado automático basada en objetivos. Si dispone de una política de escalado automático basada en objetivos, reiniciará automáticamente la capacidad deseada en función de sus reglas de escalado propias.

Puede establecer manualmente la capacidad en la consola de Amazon GameLift o mediante la AWS Command Line Interface (AWS CLI). El estado de la flota debe ser Activo.

Suspensión del escalado automático

Puede suspender toda la actividad de escalado automático de cada ubicación de la flota. Con la suspensión del escalado automático, el número deseado de instancias en la ubicación de la flota

permanece igual a menos que se cambie manualmente. Si suspende el escalado automático de una ubicación, afectará a las políticas actuales de la flota y a cualquier política que pueda definir en el futuro.

Para configurar manualmente la capacidad de la flota

Console

1. Abra la consola de Amazon GameLift.
2. En el panel de navegación, elija Alojamiento y Flotas.
3. En la página Flotas, elija el nombre de una flota activa para abrir la página de detalles de la flota.
4. En la pestaña Escalado, en Ubicaciones de escalado automático suspendidas, seleccione las ubicaciones para las que desee suspender el escalado automático y, a continuación, elija Suspender.
5. En Capacidad de escalado, seleccione una ubicación que desee configurar manualmente y, a continuación, elija Editar.
6. En el cuadro de diálogo Editar capacidad de escalado, defina el valor preferido para Instancias deseadas y, a continuación, seleccione Confirmar. Este valor indica a Amazon GameLift cuántas instancias deben mantenerse en estado activo y disponibles para alojar sesiones de juego.

   Amazon GameLift responde a los cambios mediante la implementación de instancias adicionales o el cierre de las innecesarias. A medida que Amazon GameLift completa este proceso, el número de instancias activas en la ubicación cambiará para coincidir con el valor de instancias actualizado deseado. Este proceso puede tardar cierto tiempo.

AWS CLI

1. Compruebe la configuración de la capacidad actual. En una ventana de línea de comandos, utilice el comando describe-fleet-location-capacity con el ID de la flota y la ubicación para los que quiera cambiar la capacidad. Este comando devuelve un objeto FleetCapacity que incluye la configuración de la capacidad actual de la ubicación. Determine si los límites de instancia se ajustarán a la nueva configuración de instancias deseada.

   ```
   aws gamelift describe-fleet-location-capacity \
       --fleet-id <fleet identifier> \
   ```

```
    --location <location name>
```

2. Actualice la capacidad deseada. Utilice el comando update-fleet-capacity con el ID de la flota, la ubicación y un valor nuevo de las instancias deseadas. Si este valor queda fuera del rango de límites actual, puede ajustar valores de ajuste de límites en el mismo comando.

```
--fleet-id <fleet identifier>
--location <location name>
--desired-instances <fleet capacity as an integer>
--max-size <maximum capacity>    [Optional]
--min-size <minimum capacity>    [Optional]
```

Ejemplo:

```
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 la solicitud se realiza correctamente, Amazon GameLift devuelve el ID de la flota. Si la nueva configuración de instancias deseadas está fuera de los límites máximos y mínimos, Amazon GameLift devuelve un error.

Escalado automático de la capacidad con Amazon GameLift

Utilice el escalado automático en Amazon GameLift para escalar de forma dinámica la capacidad de su flota como respuesta a la actividad del servidor de juegos. A medida que los jugadores lleguen e inicien sesiones de juego, el escalado automático puede añadir nuevas instancias, y a medida que la demanda de los jugadores se reduzca, el escalado automático irá cerrando las instancias innecesarias. El escalado automático es una forma eficaz de minimizar sus recursos y costos de alojamiento y, a la vez, facilitar al jugador una experiencia rápida y sin interrupciones.

Para usar el escalado automático, debe crear políticas de escalado que indiquen a Amazon GameLift cuándo realizar el escalado o reducción verticalmente. Existen dos tipos de políticas de escalado, las basadas en objetivos y las basadas en reglas. El enfoque basado en objetivos (el seguimiento de objetivos) es una solución completa. Lo recomendamos como la opción más sencilla y eficaz. Las

políticas de escalado basadas en reglas requieren que se defina cada aspecto del proceso de toma de decisiones de escalado automático, que son útiles para abordar cuestiones específicas. Esta solución funciona mejor como complemento del escalado automático basado en objetivos.

Puede administrar el escalado automático basado en objetivos mediante la consola de Amazon GameLift, la AWS Command Line Interface (AWS CLI) o un SDK de AWS. Puede administrar el escalado automático basado en reglas mediante la AWS CLI o un SDK de AWS, aunque puede ver las políticas de escalado basadas en reglas en la consola.

Temas

Escalado automático basado en objetivos

El escalado automático basado en objetivos para Amazon GameLift ajusta los niveles de capacidad en función de la métrica de la flota `PercentAvailableGameSessions`. Esta métrica representa el búfer disponible de la flota frente a aumentos repentinos de la demanda de los jugadores.

El principal motivo para mantener un búfer de capacidad es el tiempo de espera del jugador. Cuando las ranuras de sesiones de juego están listas y en espera, se introducen los nuevos jugadores en las sesiones de juego en cuestión de segundos. Si no hay recursos disponibles, los jugadores deben esperar a que las sesiones de juego existentes terminen o a que queden disponibles nuevos recursos. El inicio de nuevas instancias y procesos del servidor puede tardar unos minutos.

Al configurar el escalado automático basado en objetivos, especifique el tamaño del búfer que quiera que mantenga la flota. Puesto que `PercentAvailableGameSessions` mide el porcentaje de recursos disponibles, el tamaño real del búfer es un porcentaje de la capacidad total de la flota. Amazon GameLift añade o elimina instancias para mantener el tamaño del búfer de destino. Con un búfer grande, reducirá el tiempo de espera, pero también pagará por recursos adicionales que podrían no utilizarse. Si los jugadores tienen más tolerancia a los tiempos de espera, puede reducir los costos estableciendo un búfer pequeño.

Para configurar el escalado automático basado en objetivos, realice el siguiente procedimiento:

Console

1. Abra la consola de Amazon GameLift.
2. En el panel de navegación, elija Alojamiento y Flotas.

3. En la página Flotas, elija el nombre de una flota activa para abrir la página de detalles de la flota.
4. Elija la pestaña Escalado. Esta pestaña muestra las métricas históricas de escalado de la flota y contiene controles para ajustar la configuración de escalado actual.
5. En Capacidad de escalado, compruebe que los límites de Tamaño mínimo y Tamaño máximo sean los adecuados para la flota. Con el escalado automático habilitado, la capacidad se ajustará entre esos dos límites.
6. En Política de escalado automático basada en objetivos, elija Editar.
7. En el cuadro de diálogo Editar la política de escalado automático basada en objetivos, en Porcentaje de sesiones de juego disponibles, establezca el porcentaje que quiera retener y, a continuación, elija Confirmar. Tras confirmar la configuración, Amazon GameLift añadirá una nueva política basada en objetivos en Política de escalado automático basada en objetivos.

AWS CLI

1. Establezca los límites de capacidad. Configure los valores límites con el comando update-fleet-capacity. Para obtener más información, consulte Establecimiento de los límites de capacidad de Amazon GameLift.
2. Cree una política nueva. Abra una ventana de línea de comandos y utilice el comando put-scaling-policy con la configuración de parámetros de la política. Para actualizar una política existente, especifique el nombre de la política y proporcione una versión completa de la política actualizada.

```
--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>
```

Ejemplo:

```
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"
```

Escalado automático con políticas basadas en reglas

Las políticas de escalado basadas en reglas en Amazon GameLift ofrecen un control detallado al escalar automáticamente la capacidad de una flota como respuesta a la actividad de los jugadores. Para cada política, puede vincular el escalado a una métrica de la flota de entre varias, identificar un punto de desencadenamiento y personalizar el evento de ampliación o reducción de respuesta. Las políticas basadas en reglas son útiles para complementar el escalado basado en destino para afrontar circunstancias especiales.

Una política basada en reglas indica lo siguiente: "Si una métrica de una flota alcanza o supera un valor umbral durante un periodo determinado, se debe cambiar la capacidad de la flota en una cantidad determinada". En este tema se describe la sintaxis utilizada para construir una instrucción de política y proporciona ayuda para crear y administrar sus políticas basadas en reglas.

Administración de políticas basadas en reglas

Puede crear, actualizar o eliminar políticas basadas en reglas mediante el SDK de AWS o la AWS Command Line Interface (AWS CLI) con la API de servicio de Amazon GameLift. Puede ver todas las políticas activas en la consola de Amazon GameLift.

Para detener temporalmente todas las políticas de escalado para una flota, utilice el comando de la AWS CLI stop-fleet-actions.

Para crear o actualizar una política de escalado basada en reglas (AWS CLI), realice el siguiente procedimiento:

1. Establezca los límites de capacidad. Configure uno o ambos valores límites con el comando update-fleet-capacity. Para obtener más información, consulte Establecimiento de los límites de capacidad de Amazon GameLift.
2. Cree una política nueva. Abra una ventana de línea de comandos y utilice el comando put-scaling-policy con la configuración de parámetros de la política. Para actualizar una política existente, especifique el nombre de la política y proporcione una versión completa de la política actualizada.

```
--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>
```

Ejemplo:

```
aws gamelift put-scaling-policy \
    --fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \
    --name "Scale up when AGS<50" \
    --policy-type RuleBased \
    --metric-name AvailableGameSessions \
    --comparison-operator LessThanThreshold \
    --threshold 50 \
    --evaluation-periods 10 \
    --scaling-adjustment-type ChangeInCapacity \
    --scaling-adjustment 1
```

Para eliminar una política de escalado basada en reglas mediante la AWS CLI, realice el siguiente procedimiento:

- Abra una ventana de línea de comandos y utilice el comando delete-scaling-policy con el ID de la flota y el nombre de la política.

  Ejemplo:

```
aws gamelift delete-scaling-policy \
    --fleet-id fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa \
    --name "Scale up when AGS<50"
```

Sintaxis de las reglas de escalado automático

Para construir una instrucción de política de escalado automático basado en reglas, especifique seis variables:

Si *<nombre de métrica>* se mantiene en *<operador de comparación>* *<valor umbral>* durante *<periodo de evaluación>*, entonces se debe cambiar la capacidad de la flota utilizando *<tipo de ajuste>* en/por *<valor de ajuste>*.

Por ejemplo, esta instrucción de la política inicia un evento de escalado vertical cuando la capacidad adicional de la flota es inferior a lo que se necesita para administrar 50 nuevas sesiones de juego:

Si `AvailableGameSessions` se mantiene en `less than 50` durante `10 minutes`, entonces se debe cambiar la capacidad de la flota utilizando `ChangeInCapacity` en `1 instances`.

Nombre de métrica

Para iniciar un evento de escalado, vincule una política de escalado automático a una de las siguientes métricas específicas de la flota. Consulte Métricas de Amazon GameLift para flotas para obtener descripciones completas de las métricas.

- Activación de sesiones de juego
- Sesiones de juego activas
- Sesiones de juego disponibles
- Porcentaje de sesiones de juego disponibles
- Instancias activas
- Sesiones de jugador disponibles
- Sesiones de jugador actuales
- Instancias inactivas
- Porcentaje de instancias inactivas

Si la flota se encuentra en una cola de sesión de juego, puede utilizar las siguientes métricas:

- Profundidad de la cola: número de solicitudes de sesiones de juego pendientes para las que esta flota es la mejor ubicación de alojamiento disponible.
- Tiempo de espera: tiempo de espera específico de la flota. Periodo de tiempo que ha estado esperando la solicitud de sesión de juego más antigua hasta ser completada. El tiempo de espera de una flota es igual al tiempo en cola de la solicitud actual más vieja.

Operador de comparación

Indica a Amazon GameLift cómo debe comparar los datos de las métricas con el valor de umbral. Los operadores de comparación válidos incluyen mayor que (**>**), menor que (**<**), mayor o igual que (**>=**) y menor o igual que (**<=**).

Valor umbral

Cuando el valor especificado de la métrica alcanza o supera el valor umbral, inicia un evento de escalado. Este valor siempre es un número entero positivo.

Periodo de evaluación

La métrica debe alcanzar o superar el valor umbral durante la totalidad del periodo de evaluación antes de iniciar un evento de escalado. La longitud del periodo de evaluación es consecutiva: si la métrica cae por debajo del umbral, el periodo de evaluación se inicia de nuevo.

Tipo y valor de ajuste

Este conjunto de variables opera conjuntamente para especificar el modo en que Amazon GameLift debe ajustar la capacidad de la flota cuando se inicia un evento de escalado. Puede elegir de entre tres posibles tipos de ajuste:

- Cambio de capacidad: permite aumentar o disminuir la capacidad actual según el número de instancias especificado. Especifique el valor de ajuste según el número de instancias que desea añadir o quitar de la flota. Los valores positivos añaden instancias, mientras que los valores negativos eliminan instancias. Por ejemplo, un valor de "-10" reducirá verticalmente la flota en 10 instancias, independientemente del tamaño total de la flota.
- Cambio porcentual de la capacidad: permite aumentar o disminuir la capacidad actual según el porcentaje especificado. Especifique el valor de ajuste en función del porcentaje en que desea aumentar o reducir la capacidad de la flota. Los valores positivos añaden instancias, mientras que los valores negativos eliminan instancias. Por ejemplo, en el caso de una flota con 50 instancias, un cambio porcentual de "20" añadirá diez instancias a la flota.
- Capacidad exacta: permite aumentar o reducir la capacidad actual a un valor específico. Especifique el valor de ajuste en el número de instancias exacto que desea mantener en la flota.

Consejos para el escalado automático basado en reglas

Las siguientes sugerencias pueden ayudarle a sacar el máximo provecho del escalado automático con políticas basadas en reglas.

Utilice varias políticas

Puede emplear varias políticas de escalado automático para una flota al mismo tiempo. La situación más habitual consiste en tener una política basada en destino que administre la mayoría de las necesidades de escalado y utilizar políticas basadas en reglas para administrar casos extremos. No existen límites en cuanto al uso de varias políticas.

Con varias políticas, cada política se comporta de forma independiente. No se puede controlar la secuencia de eventos de escalado. Por ejemplo, si dispone de varias políticas de escalado vertical,

es posible que la actividad de los jugadores inicie varios eventos de escalado de forma simultánea. Evite las políticas que se inician entre sí. Por ejemplo, puede crear un bucle infinito si crea políticas de escalado y reducción verticales que establecen una capacidad superior al umbral de cada una.

Defina una capacidad máxima y mínima

Cada flota tiene una límite de capacidad máxima y mínima. Esta característica es especialmente importante cuando se utiliza el escalado automático. El escalado automático nunca definirá un valor de la capacidad que no esté dentro de ese rango. De forma predeterminada, las flotas que se acaban de crear tienen un mínimo de 0 y un máximo de 1. Para que la política de escalado automático afecte a la capacidad según lo previsto, aumente el valor máximo.

La capacidad de la flota también está acotada por los límites del tipo de instancia de la flota y por las cuotas de servicio de su Cuenta de AWS. No puede establecer un valor mínimo y uno máximo que se encuentren fuera de esos límites y de las cuotas de cuenta.

Realice el seguimiento de las métricas después de un cambio en la capacidad

Después de cambiar la capacidad como respuesta a una política de escalado automático, Amazon GameLift espera diez minutos para responder a los desencadenadores de dicha política. Esta espera proporciona a Amazon GameLift tiempo para agregar instancias nuevas, lanzar los servidores de juegos, conectar a los jugadores y empezar a recopilar datos de las instancias nuevas. Durante este tiempo, Amazon GameLift evalúa la política en función de la métrica y realiza el seguimiento del periodo de evaluación de la política, que se reinicia después de que se produzca un evento de escalado. Esto significa que una política de escalado podría iniciar otro evento de escalado en cuanto acabe el tiempo de espera.

No hay tiempo de espera entre los eventos de escalado que inician diferentes políticas de escalado automático.

Configuración de colas de Amazon GameLift para la ubicación de las sesiones de juego

Una cola de sesiones de juego es el mecanismo principal para procesar las nuevas solicitudes de sesiones de juego y localizar los servidores de juegos disponibles para alojarlas. Las colas ofrecen importantes ventajas a los desarrolladores de juegos y a los jugadores. Entre ellas se incluyen:

- Las colas ofrecen la mejor ubicación posible. Al procesar las solicitudes de ubicación de las sesiones de juego, una cola utiliza los algoritmos de Amazon GameLift para priorizar las ubicaciones de las colas en función de un conjunto de preferencias definidas.
- Aloje juegos en flotas de spot de menor precio. Utilice las colas para optimizar el uso de las flotas de spot de AWS que ofrecen unos costos de alojamiento considerablemente más bajos. De forma predeterminada, las colas siempre intentan incluir nuevas sesiones de juego en las flotas de spot.
- Ubique nuevos juegos más rápido cuando haya mucha demanda. Las colas utilizan varias ubicaciones posibles para las ubicaciones. Esto significa que siempre hay capacidad de reserva si la ubicación de colocación preferida no está disponible.
- Aumento de la resiliencia de la disponibilidad de los juegos. Se pueden producir cortes de energía. Cuando se utiliza una cola de varias regiones, una ralentización o una interrupción no tiene por qué afectar al acceso de los jugadores al juego.
- Uso más eficaz de la capacidad adicional de las flotas. Para hacer frente a aumentos inesperados en la demanda de jugadores, las colas ofrecen un acceso rápido a la capacidad de alojamiento adicional. Las ubicaciones de la flota en una cola proporcionan una capacidad de respaldo mutua. Las ubicaciones se escalan o reducen verticalmente en función de la demanda de los jugadores.
- Obtención de métricas de las ubicaciones de sesiones de juego y del desempeño de las colas. Amazon GameLift emite métricas de colas, que incluyen estadísticas sobre éxitos y errores de ubicación, el número de solicitudes en la cola y el tiempo medio que las solicitudes pasan en la cola. Puede ver estas métricas en la consola de Amazon GameLift o en CloudWatch.

Para empezar a trabajar rápidamente con colas, consulte Diseño de colas de sesiones de juego.

Diseño de colas de sesiones de juego

En este tema se describe cómo diseñar una cola que ofrezca una experiencia de jugador con una latencia mínima y que utilice los recursos de alojamiento de forma eficiente. Para obtener más información sobre las colas de sesiones de juego y su funcionamiento, consulte Configuración de colas de Amazon GameLift para la ubicación de las sesiones de juego.

Estas características de Amazon GameLift requieren colas:

- Emparejamientos con FlexMatch
- Usa instancias puntuales con Amazon GameLift

Definición del ámbito de la cola

Es posible que la población de jugadores del juego tenga grupos de jugadores que no deberían jugar juntos. Por ejemplo, si publica el juego en dos idiomas, cada idioma debería tener sus propios servidores de juegos.

Para configurar la ubicación de las sesiones de juego para su población de jugadores, cree una cola independiente para cada segmento de jugadores. Revise cada cola para ubicar a los jugadores en los servidores de juegos correctos. Algunas formas habituales de determinar el ámbito de las colas son las siguientes:

- Por ubicaciones geográficas. Al implementar los servidores de juegos en varias áreas geográficas, puede crear colas para los jugadores en cada ubicación con el fin de reducir la latencia de los jugadores.
- Por compilación o variaciones del script. Si tiene más de una variante del servidor de juegos, es posible que esté ofreciendo soporte a grupos de jugadores que no puedan jugar en las mismas sesiones de juego. Por ejemplo, es posible que las compilaciones o scripts de los servidores de juegos admitan distintos idiomas o tipos de dispositivos.
- Por tipos de eventos. Puede crear una cola especial para administrar los juegos de los participantes en torneos u otros eventos especiales.

Creación de una política de latencia de jugadores

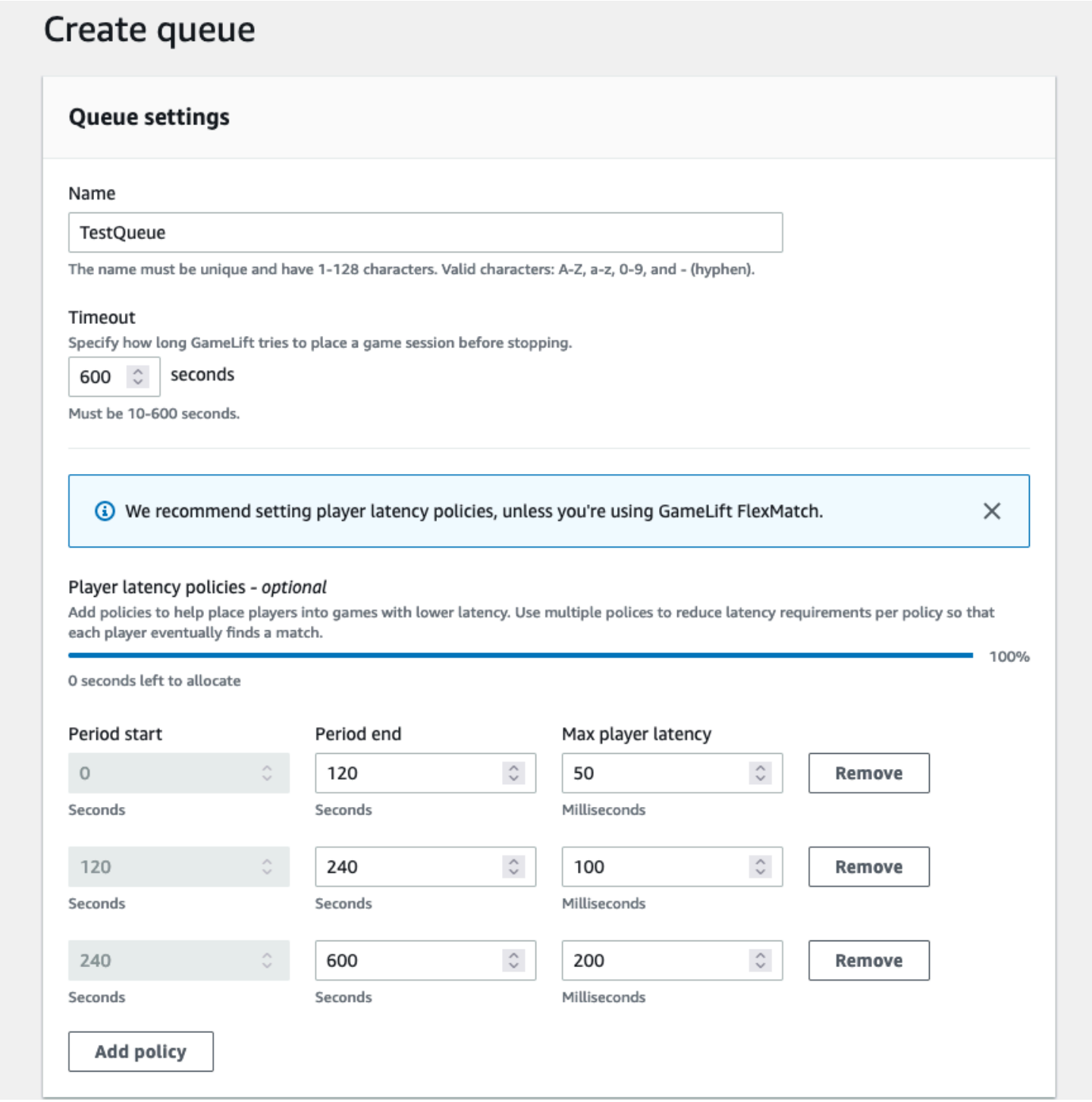
Si las solicitudes de posicionamiento incluyen datos de latencia de los jugadores, el algoritmo busca las sesiones de juego en las ubicaciones con la latencia media más baja de todos los jugadores. Ubicar las sesiones de juego en función de la latencia media de los jugadores impide que Amazon GameLift ubique a la mayoría de los jugadores en juegos con una latencia alta. Sin embargo, Amazon GameLift sigue ubicando a los jugadores con una latencia extrema. Para adaptarse a estos jugadores, cree políticas de latencia para los jugadores.

Una política de latencia de jugadores impide que Amazon GameLift ubique una sesión de juego solicitada en cualquier lugar en el que los jugadores de la solicitud experimenten una latencia superior al valor máximo. Las políticas de latencia de los jugadores también pueden impedir que Amazon GameLift haga coincidir las solicitudes de sesión de juego con las de los jugadores con una latencia más alta.

> **Tip**
>
> Para administrar reglas específicas de latencia, como exigir una latencia similar a todos los jugadores de un grupo, puede utilizar Amazon GameLift FlexMatch para crear reglas de emparejamiento basadas en la latencia.

Por ejemplo, considere esta cola con un tiempo de espera de 5 minutos y las siguientes políticas de latencia de los jugadores:

1. Si se pasa 120 segundos buscando una ubicación donde todas las latencias de los jugadores son de menos de 50 milisegundos.
2. Si se pasa 120 segundos buscando una ubicación donde todas las latencias de los jugadores son de menos de 100 milisegundos.
3. Si se emplea el tiempo de la cola restante en buscar una ubicación donde todas las latencias de los jugadores son de menos de 200 milisegundos.



Creación una cola con varias ubicaciones

Recomendamos un diseño con varias ubicaciones para todas las colas. Este diseño puede mejorar la velocidad de ubicación y la resiliencia del alojamiento. Se requiere un diseño de varias ubicaciones para utilizar los datos de latencia de los jugadores con el fin de que puedan participar en las sesiones de juego con una latencia mínima. Si va a crear colas en varias ubicaciones que utilizan flotas de

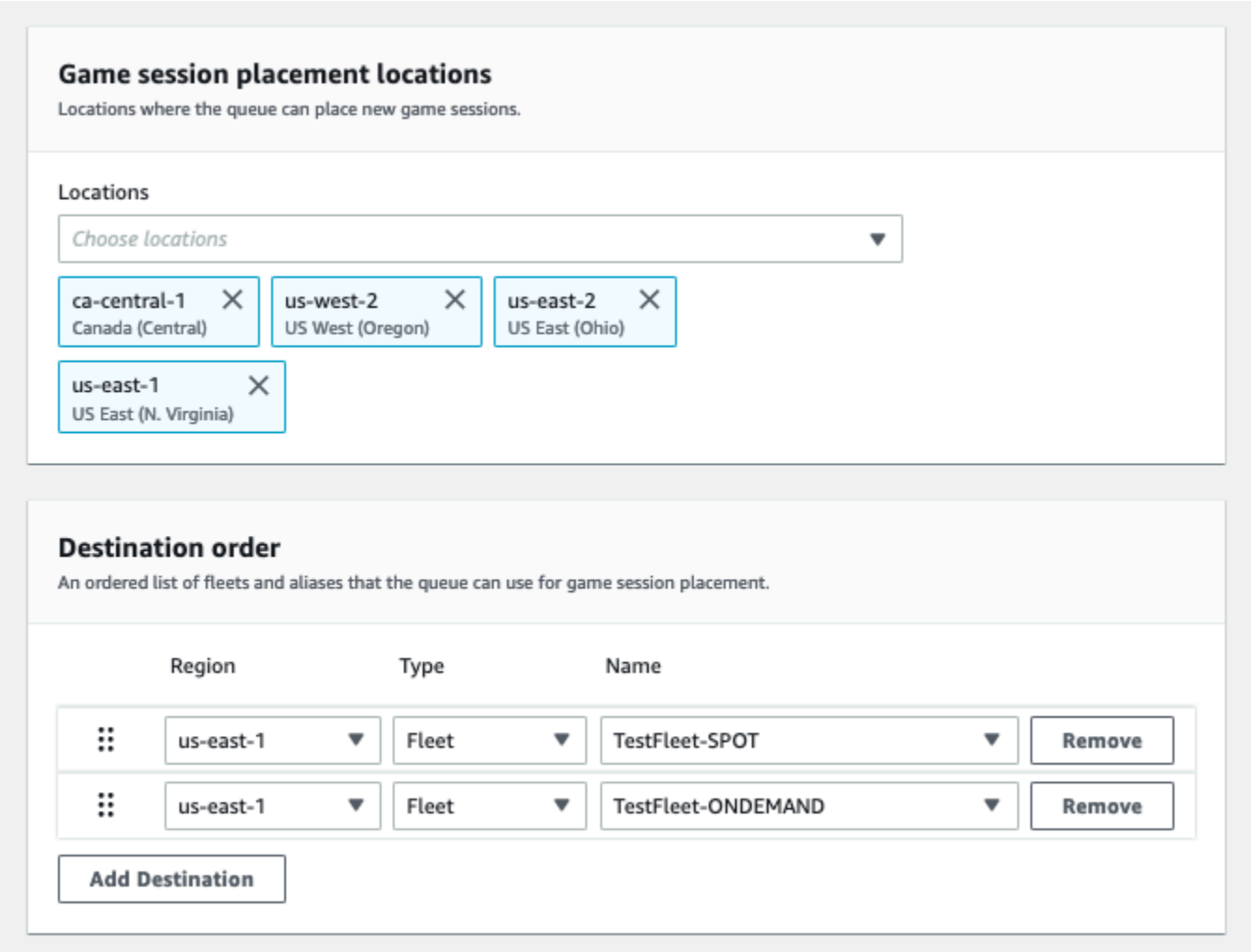
instancias de spot, siga las instrucciones que se indican en Tutorial: Configuración de una cola de sesiones de juego para instancias de spot.

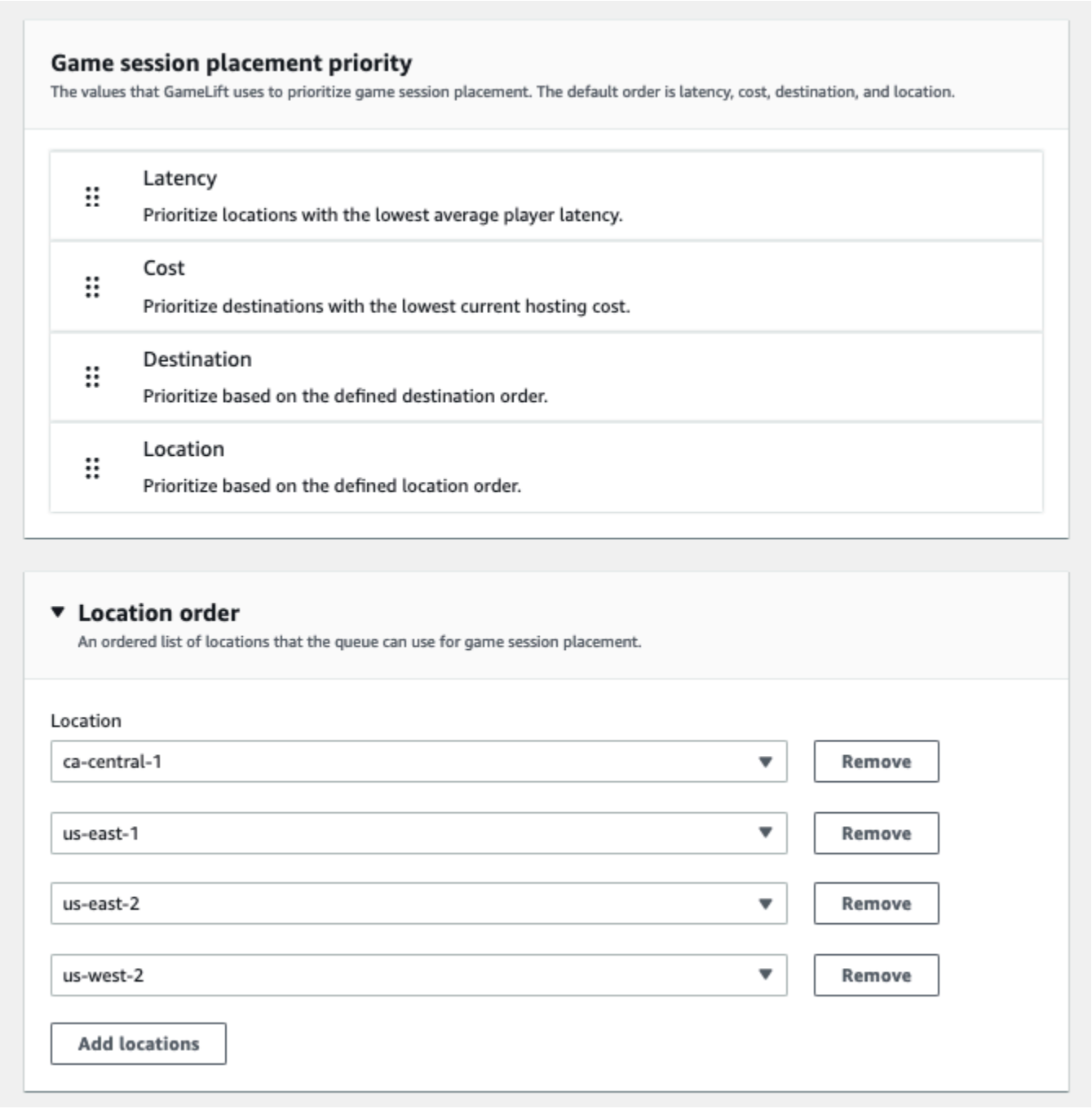
Una forma de crear una cola con varias ubicaciones consiste en añadir una flota con varias ubicaciones a una cola. De esa forma, la cola puede colocar las sesiones de juego en cualquiera de las ubicaciones de la flota. También puede añadir otras flotas con diferentes configuraciones o ubicaciones de origen para aumentar la redundancia. Si utiliza una flota de instancias de spot con varias ubicaciones, siga las prácticas recomendadas e incluya una flota de instancias bajo demanda con las mismas ubicaciones.

En el siguiente ejemplo, se describe el proceso de diseño de una cola básica con varias ubicaciones. En este ejemplo, utilizamos dos flotas: una flota de instancias de spot y una flota de instancias bajo demanda. Cada flota tiene las siguientes Regiones de AWS para las ubicaciones: `us-east-1`, `us-east-2ca-central-1` y `us-west-2`.

Para crear una cola básica de varias ubicaciones con flotas de varias ubicaciones, realice el siguiente procedimiento:

1. Elija una ubicación para crear la cola. Puede minimizar la latencia de las solicitudes colocando la cola en una ubicación cercana a donde implementó el servicio de cliente. En este ejemplo, crearemos la cola en `us-east-1`.
2. Cree una cola nueva y añada nuestras flotas de varias ubicaciones como destinos de la cola. El orden de destino determina la forma en que Amazon GameLift ubica las sesiones de juego. En este ejemplo, incluimos primero la flota de instancias de spot y, en segundo lugar, la flota de instancias bajo demanda.
3. Defina el orden de prioridad de la ubicación de las sesiones de juego de la cola. Este orden determina dónde busca primero la cola un servidor de juegos disponible. En este ejemplo, utilizaremos el orden de prioridad predeterminado.
4. Defina el orden de ubicación. Si no define el orden de las ubicaciones, Amazon GameLift utilizará las ubicaciones en orden alfabético.


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



Priorice la ubicación de las sesiones de juego.

Amazon GameLift utiliza el algoritmo de FleetIQ para determinar dónde ubicar una nueva sesión de juego en función de un conjunto ordenado de criterios. Puede utilizar el orden de prioridad predeterminado o personalizar el orden.

Orden de prioridad predeterminado

Para las solicitudes de ubicación que incluyen datos de latencia de los jugadores, FleetIQ prioriza los criterios de ubicación de la sesión de juego en el siguiente orden predeterminado:

1. Latencia: la latencia media más baja para todos los jugadores de la solicitud.
2. Costo: el costo de alojamiento más bajo, si la latencia es igual en varias ubicaciones. El costo del alojamiento se basa principalmente en una combinación del tipo de instancia y ubicación.
3. Destino: pedido de destino, si la latencia y el costo son iguales en varias ubicaciones. FleetIQ prioriza los destinos según el orden que aparece en la configuración de la cola.
4. Ubicación: orden de ubicación, si la latencia, el costo y el destino son iguales en varias ubicaciones. FleetIQ prioriza los destinos según las ubicaciones que aparecen en la configuración de la cola.

Orden de prioridad personalizado

Para personalizar el orden de prioridad de una cola en la consola de Amazon GameLift, arrastre el valor de prioridad a la posición en la que desee ubicarlo. Para personalizar el orden de prioridad de una cola mediante AWS Command Line Interface (AWS CLI), utilice el comando create-game-session-queue junto con la opción --priority-configuration. Puede utilizar este comando para crear una nueva cola o para actualizar una cola existente.

El algoritmo de FleetIQ añade cualquier criterio que no se mencione explícitamente al final de la lista, según el orden predeterminado. Si incluye el criterio de ubicación en la configuración de prioridad, también debe proporcionar una lista ordenada de ubicaciones.

Diseño de varias colas según sea necesario

En función del juego y de los jugadores, es posible que desee crear más de una cola de sesión de juego. Cuando su servicio de cliente de juegos solicite una nueva sesión de juego, especifica qué cola de sesión de juego utilizar. Para ayudarle a determinar si debe usar varias colas, tenga en cuenta los siguientes aspectos:

- Variaciones del servidor de juegos. Puede crear una cola independiente para cada variación del servidor de juegos. Todas las flotas de una cola deben implementar servidores de juegos compatibles. Esto se debe a que los jugadores que utilicen la cola para unirse a los juegos deben poder jugar en cualquiera de los servidores de juegos de la cola.
- Diferentes grupos de jugadores. Puede personalizar la forma en que Amazon GameLift ubica las sesiones de juego en función del grupo de jugadores. Por ejemplo, es posible que necesite personalizar las colas para determinados modos de juego que requieren un tipo de instancia

especial o una configuración de tiempo de ejecución. O bien, puede que le interese una cola especial para administrar las posiciones de un torneo u otro evento.

- Métricas de las colas de las sesiones de juego. Puede configurar las colas en función de cómo desee recopilar las métricas de ubicación de las sesiones de juego. Para obtener más información, consulte Métricas de Amazon GameLift para colas.

Evaluación de métricas de cola

Utilice métricas para evaluar el desempeño de las colas. Puede ver las métricas relacionadas con las colas en la consola de Amazon GameLift o en Amazon CloudWatch. Para ver una lista y descripciones de métricas de colas, consulte Métricas de Amazon GameLift para colas.

Las métricas de colas pueden proporcionar información sobre los siguientes aspectos:

- Rendimiento general de la cola: las métricas de la cola indican el grado de éxito con el que una cola responde a las solicitudes de ubicación. Esas métricas también pueden ayudarle a identificar cuándo y por qué fallan las ubicaciones. En el caso de las colas con flotas con escalación manual, las métricas de `AverageWaitTime` y `QueueDepth` pueden indicar cuándo debe ajustar la capacidad de una cola.
- Rendimiento del algoritmo de FleetIQ: para las solicitudes de ubicación que utilizan el algoritmo de FleetIQ, las métricas muestran la frecuencia con la que el algoritmo encuentra la ubicación ideal para las sesiones de juego. La ubicación puede priorizar el uso de los recursos con la menor latencia de jugadores o los recursos con el menor costo. También hay métricas de error que identifican los motivos más comunes por los que Amazon GameLift no encuentra una ubicación ideal. Para obtener más información sobre las métricas, consulte Supervisión de Amazon GameLift con Amazon CloudWatch.
- Ubicaciones específicas por ubicación: en el caso de las colas en varias ubicaciones, las métricas muestran las ubicaciones correctas por ubicación. En el caso de las colas que utilizan el algoritmo de FleetIQ, estos datos proporcionan información útil sobre de dónde se produce la actividad de los jugadores.

Cuando evalúe las métricas de rendimiento del algoritmo de FleetIQ, tenga en cuenta las siguientes recomendaciones:

- Para realizar un seguimiento de la tasa de búsqueda de una ubicación ideal en la cola, utilice la métrica `PlacementsSucceeded` en combinación con las métricas de FleetIQ para obtener la latencia más baja y el precio más bajo.

- Para aumentar la tasa de búsqueda de una ubicación ideal en una cola, revise las siguientes métricas de error:
  - Si el valor de `FirstChoiceOutOfCapacity` es alto, ajuste la escala de capacidad para las flotas de la cola.
  - Si la métrica de error `FirstChoiceNotViable` es alta, compruebe las flotas de instancias de spot. Las flotas de instancias de spot no se consideran viables cuando la tasa de interrupción es muy alta en un determinado tipo de instancia. Para solucionar este problema, cambie la cola para que utilice flotas de instancias de spot con otros tipos de instancias. Le recomendamos que incluya flotas de instancias de spot con diferentes tipos de instancias en cada ubicación.

Prácticas recomendadas para las colas de sesiones de juego de Amazon GameLift

Estas son algunas de las prácticas recomendadas que pueden ayudarle a crear colas de sesiones de juego eficaces para ubicar las sesiones de juego.

Prácticas recomendadas para colas con cualquier tipo de flota

Una cola contiene una lista de destinos de flota donde se pueden ubicar nuevas sesiones de juego. Cada flota puede tener instancias implementadas en varias ubicaciones geográficas. Al elegir una ubicación, la cola selecciona una combinación de una flota y una ubicación de la flota. Proporcione un conjunto de prioridades para que la cola las utilice al elegir una ubicación.

Tenga en cuenta las siguientes directrices y prácticas recomendadas:

- Añada flotas en ubicaciones que incluyan a sus jugadores. Puede añadir flotas y alias en cualquier ubicación disponible. La ubicación es importante si realiza ubicaciones en función de la latencia informada de los jugadores.
- Utilice alias para todas las flotas. Asigne un alias a cada flota en una cola y utilice los nombres de los alias al configurar los destinos en la cola.
- Utilice las mismas compilaciones del juego o scripts, o similares, para todas las flotas. La cola puede incluir a los jugadores en sesiones de juego de cualquier flota en la cola. Los jugadores deben poder jugar en cualquier sesión de juego de cualquier flota.
- Cree flotas en al menos dos ubicaciones. Al tener los servidores de juegos alojados en al menos otra ubicación, se reducirá el impacto de las interrupciones regionales en sus jugadores. Puede

mantener sus flotas de respaldo con una reducción vertical y usar el escalado automático para aumentar la capacidad si el uso aumenta.

- Priorice la ubicación de las sesiones de juego. Una cola prioriza las opciones de ubicación en función de varios elementos, incluido el orden de la lista de destinos.
- Cree la cola en la misma ubicación que el servicio de cliente. Al colocar la cola en una ubicación cercana al servicio de cliente, puede minimizar la latencia de la comunicación.
- Utilice flotas con varias ubicaciones. Utilice la configuración del filtro de colas para evitar que la cola coloque las sesiones de juego en ubicaciones específicas. Puede utilizar al menos dos flotas con varias ubicaciones con diferentes ubicaciones de origen para reducir el impacto de las ubicaciones de los juegos durante una interrupción regional.
- Utilice la misma configuración de certificado TLS para todas las flotas. Los clientes de juego que se conectan a las sesiones de juego de sus flotas deben tener protocolos de comunicación compatibles.

Prácticas recomendadas para colas con flotas de spot

Si su cola incluye flotas de spot, configure una cola resiliente. De esa forma, se aprovecha el ahorro de costos de las flotas de spot y, al mismo tiempo, se minimiza el efecto de las interrupciones de las sesiones de juego. Si necesita ayuda para crear correctamente las flotas y las colas de las sesiones de juego para usarlas con las flotas de spot, consulte Tutorial: Configuración de una cola de sesiones de juego para instancias de spot. Para obtener más información sobre las instancias de spot, consulte Usa instancias puntuales con Amazon GameLift.

Además de las prácticas recomendadas generales de la sección anterior, tenga en cuenta estas prácticas recomendadas específicas de spot:

- Cree al menos una flota bajo demanda en cada ubicación. Las flotas bajo demanda proporcionan servidores de juegos de respaldo para sus jugadores. Puede mantener sus flotas de respaldo reducidas verticalmente hasta que las necesite y utilizar el escalado automático para aumentar la capacidad bajo demanda cuando las flotas de spot no estén disponibles.
- Seleccione diferentes tipos de instancias en varias flotas de spot en una ubicación. Si un tipo de instancia de spot deja de estar disponible temporalmente, la interrupción solo afectará a una flota de spot de la ubicación. La práctica recomendada es elegir tipos de instancias ampliamente disponibles y utilizar tipos de instancias de la misma familia (por ejemplo, m5.large, m5.xlarge y m5.2xlarge). Utilice la consola de Amazon GameLift para ver los datos históricos de precios de los tipos de instancias.

Creación de una cola de sesión de juego

Las colas se utilizan para ubicar nuevas sesiones de juego con los mejores recursos de alojamiento disponibles en varias flotas y regiones. Para obtener más información sobre la creación de colas para un juego, consulte Diseño de colas de sesiones de juego.

En un cliente de juego, las sesiones de juego nuevas se inician en las colas mediante las solicitudes de ubicación. Para obtener más información sobre la ubicación de sesiones de juego, consulte Creación de sesiones de juego.

Al actualizar el destino de una cola, hay un breve periodo de transición (hasta 30 segundos) durante el que las sesiones de juego incluidas en los destinos de la cola pueden terminar en la flota anterior.

Console

1. En la consola de Amazon GameLift, en la página de navegación, elija Colas.
2. En la página Queues (Colas), elija Create queue (Crear nueva cola).
3. En la página Crear cola, en Configuración de cola, realice el siguiente procedimiento:

    a. En Nombre, escriba un nombre de cola.

    b. En Tiempo de espera, especifique el tiempo que desea que Amazon GameLift intente ubicar una sesión de juego antes de detenerla. Amazon GameLift continuará buscando recursos disponibles en cualquier flota hasta que se agote el tiempo de la solicitud.

    c. Para las políticas de latencia de los jugadores, introduzca durante cuánto tiempo Amazon GameLift debe buscar recursos dentro de la latencia máxima definida (opcional). Añada políticas adicionales para flexibilizar gradualmente la latencia máxima. Para añadir políticas adicionales, seleccione Añadir política.
4. En Ubicaciones de colocación de sesiones de juego, seleccione las ubicaciones para incluirlas en la lista. De forma predeterminada, se incluyen Todas las ubicaciones. Todas las flotas de la cola deben tener la misma configuración de certificado. Todas las flotas deberían tener compilaciones de juego compatibles con los clientes de juego que utilizan la cola.
5. En Pedido de destino, añada uno o varios destinos a la cola.

    a. Elija Add destination.

    b. Seleccione la ubicación en la que se encuentra el destino.

    c. Seleccione el tipo de destino.

    d. En la lista de nombres de flota o alias resultante, seleccione el que desea añadir.

e. Si dispone de varios destinos, establezca el pedido predeterminado arrastrando el icono de los seis puntos a la izquierda del destino. Amazon GameLift utiliza este pedido al buscar destinos para que los recursos disponibles ubiquen una sesión de juego nueva.

6. Para establecer la Prioridad de ubicación de las sesiones de juego, añada y arrastre los valores de Latencia, Costo, Destino y Ubicación para definir cómo Amazon GameLift prioriza las flotas de la cola. Para obtener más información sobre la priorización de flotas, consulte Priorice la ubicación de las sesiones de juego..
7. Añada ubicaciones a su orden de ubicación y arrástrelas hasta la prioridad que debe utilizar la cola. Si la ubicación es la última prioridad a la hora de ubicar una sesión de juego, Amazon GameLift la utiliza como factor de desempate.
8. En Configuración de notificaciones de eventos, haga lo siguiente (opcional):

   a. Seleccione o cree un tema de SNS para recibir notificaciones de eventos relacionadas con la ubicación. Para obtener más información sobre las notificaciones de eventos de , consulte .

   b. Añada Datos de eventos personalizados para añadirlos a eventos creados por esta cola.
9. Añada Etiquetas (opcional). Para obtener más información sobre el etiquetado, consulte Etiquetado de recursos de AWS.
10. Seleccione Create (Crear).

AWS CLI

Example Creación de una cola

En el ejemplo siguiente se crea una cola de sesión de juego con estas configuraciones:

- Un tiempo de espera de cinco minutos
- Dos destinos de flota
- Filtros para permitir solo ubicaciones en `us-east-1`, `us-east-2`, `us-west-2` y `ca-central-1`
- Prioriza los destinos en función del costo y, después, de las ubicaciones en el orden definido.

```
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**
>
> Puede obtener los valores de los ARN de la flota y del alias llamando a describe-fleet-attributes o a describe-alias con el ID de la flota o del alias.

Si la `create-game-session-queue` solicitud se realiza correctamente, Amazon GameLift devuelve un objeto GameSessionQueue con la configuración de la cola nueva. Ahora puede enviar solicitudes a la cola utilizando StartGameSessionPlacement.

Example Creación de una cola con las políticas de latencia de los jugadores

En el ejemplo siguiente se crea una cola de sesión de juego con estas configuraciones:

- Un tiempo de espera de diez minutos
- Tres destinos de flota
- Un conjunto de políticas de latencia de los jugadores

```
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` solicitud se realiza correctamente, Amazon GameLift devuelve un objeto GameSessionQueue con la configuración de la cola nueva.

Configuración de la notificación de eventos para la ubicación de sesiones de juego.

Puede utilizar notificaciones de eventos para supervisar el estado de las solicitudes de ubicación individuales. Le recomendamos configurar las notificaciones de eventos para todos los juegos con un gran volumen de actividad de ubicación.

Existen dos opciones para configurar las notificaciones de eventos.

- Haga que Amazon GameLift publique notificaciones de eventos en un tema de Amazon Simple Notification Service (Amazon SNS) mediante una cola.
- Utilice los eventos de Amazon EventBridge publicados automáticamente y su conjunto de herramientas para administrar los eventos.

Para obtener una lista de los eventos de ubicación de sesiones de juego emitidos por Amazon GameLift, consulte Eventos de ubicación de sesión de juego.

Configuración de un tema de SNS

Para que Amazon GameLift publique todos los eventos generados por una cola de sesiones de juego en un tema, defina el campo de destino de notificaciones en él.

Para configurar un tema de SNS para la notificación de eventos de Amazon GameLift, realice el siguiente procedimiento:

1. Inicie sesión en AWS Management Console y abra la consola de Amazon SNS en https://console.aws.amazon.com/sns/v3/home.
2. En la página Temas de SNS, elija Crear tema y siga las instrucciones para crear el tema.
3. En Política de acceso, haga lo siguiente:

    a. Elija el método Avanzado.

b. Añada la siguiente sección en negrita del objeto JSON a la política existente.

```
{
  "Version": "2008-10-17",
  "Id": "__default_policy_ID",
  "Statement": [
    {
      "Sid": "__default_statement_ID",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "SNS:GetTopicAttributes",
        "SNS:SetTopicAttributes",
        "SNS:AddPermission",
        "SNS:RemovePermission",
        "SNS:DeleteTopic",
        "SNS:Subscribe",
        "SNS:ListSubscriptionsByTopic",
        "SNS:Publish"
      ],
      "Resource": "arn:aws:sns:your_region:your_account:your_topic_name",
      "Condition": {
        "StringEquals": {
          "AWS:SourceAccount": "your_account"
        }
      }
    },
    {
      "Sid": "__console_pub_0",
      "Effect": "Allow",
      "Principal": {
        "Service": "gamelift.amazonaws.com"
      },
      "Action": "sns:Publish",
      "Resource": "arn:aws:sns:your_region:your_account:your_topic_name",
      "Condition": {
        "ArnLike": {
          "aws:SourceArn":
 "arn:aws:gamelift:your_region:your_account:gamesessionqueue/your_queue_name"
        }
      }
```

```
    }
  ]
}
```

c. Añada un control de acceso adicional al tema agregando condiciones a la política de recursos (opcional).

4. Elija Create new topic (Crear nuevo tema).
5. Una vez que haya creado su tema de SNS, añádalo a las colas durante la creación de las colas o edite una cola existente para agregarlo.

Configuración de un tema de SNS con cifrado del servidor

Con el cifrado del servidor (SSE), puede almacenar información confidencial en temas cifrados. SSE protege el contenido de los mensajes en temas de Amazon SNS mediante claves que se administran en AWS Key Management Service (AWS KMS). Para obtener más información sobre el cifrado del lado del servidor con Amazon SNS, consulte Cifrado en reposo en la Guía para desarrolladores de Amazon Simple Notification Service.

Para configurar un tema de SNS con cifrado del lado del servidor, revise los temas siguientes:

- Creación de clave en la Guía para desarrolladores de AWS Key Management Service
- Habilitación de SSE para un tema en la Guía para desarrolladores de Amazon Simple Notification Service

Al crear la clave de KMS, utilice la siguiente política de claves de KMS:

```
{
    "Effect": "Allow",
    "Principal": {
        "Service": "gamelift.amazonaws.com"
     },
    "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
    ],
    "Resource": "*",
    "Condition": {
        "ArnLike": {
            "aws:SourceArn":
 "arn:aws:gamelift:your_region:your_account:gamesessionqueue/your_queue_name"
```

```
        },
        "StringEquals": {
            "kms:EncryptionContext:aws:sns:topicArn":
 "arn:aws:sns:your_region:your_account:your_sns_topic_name"
        }
    }
}
```

Configuración de EventBridge

Amazon GameLift publica automáticamente todos los eventos de ubicación de las sesiones de juego en EventBridge. Con EventBridge, puede configurar reglas para que los eventos se envíen a los destinos para su procesamiento. Por ejemplo, puede establecer una regla para dirigir el evento `PlacementFulfilled` a una función AWS Lambda que se encargue de las tareas previas a la conexión a una sesión de juego. Para obtener más información acerca de EventBridge, consulte ¿Qué es Amazon EventBridge? en la Guía del usuario de Amazon EventBridge.

A continuación se muestran algunos ejemplos de reglas de EventBridge que se pueden utilizar con las colas de Amazon GameLift:

Coincidencia de eventos de todas las colas de Amazon GameLift

```
{
    "source": [
        "aws.gamelift"
    ],
    "detail-type": [
        "GameLift Queue Placement Event"
    ]
}
```

Coincidencia de eventos de una cola específica

```
{
    "source": [
        "aws.gamelift"
    ],
    "detail-type": [
        "GameLift Queue Placement Event"
    ],
    "resources": [
        "arn:aws:gamelift:your_region:your_account:gamesessionqueue/your_queue_name"
```

```
    ]
}
```

Tutorial: Configuración de una cola de sesiones de juego para instancias de spot

Introducción

En este tutorial se describe cómo configurar la ubicación de las sesiones de juego para los juegos implementados en flotas de spot de bajo costo. Las flotas de spot requieren medidas adicionales para mantener la disponibilidad continua de los servidores de juego para los jugadores.

Destinatarios previstos

Este tutorial está dirigido a los desarrolladores de juegos que quieran utilizar las flotas de spot para alojar servidores de juegos personalizados o Realtime Servers.

Aprenderá a lo siguiente:

- Definir el grupo de jugadores al que sirve la cola de la sesión de juego.
- Construir una infraestructura de flota que respalde el ámbito de la cola de sesiones de juego.
- Asignar un alias a cada flota para abstraer el ID de la flota.
- Crear una cola, añadir flotas y priorizar dónde coloca Amazon GameLift las sesiones de juego.
- Añadir políticas de latencia de los jugadores para ayudar a minimizar los problemas de latencia.

Requisitos previos

Antes de crear flotas y colas para ubicar en las sesiones de juego, realice las siguientes tareas:

- Consulte Cómo funciona Amazon GameLift.
- Integre el servidor de juegos con Amazon GameLift.
- Cargue la compilación del servidor de juegos o el script de Realtime en Amazon GameLift.
- Planifique la configuración de su flota.

Paso 1: Definición del ámbito de la cola

En este tutorial, diseñaremos una cola para un juego que tiene una variación de compilación de servidor de juegos. En el momento del lanzamiento, publicaremos el juego en dos ubicaciones: Asia-Pacífico (Seúl) y Asia-Pacífico (Singapur). Como esas ubicaciones están cerca, la latencia no es un problema para nuestros jugadores.

En este ejemplo, hay un segmento de jugadores, lo que significa que crearemos una cola. En el futuro, cuando lancemos el juego en Norteamérica, podremos crear una segunda cola destinada a los jugadores de Norteamérica.

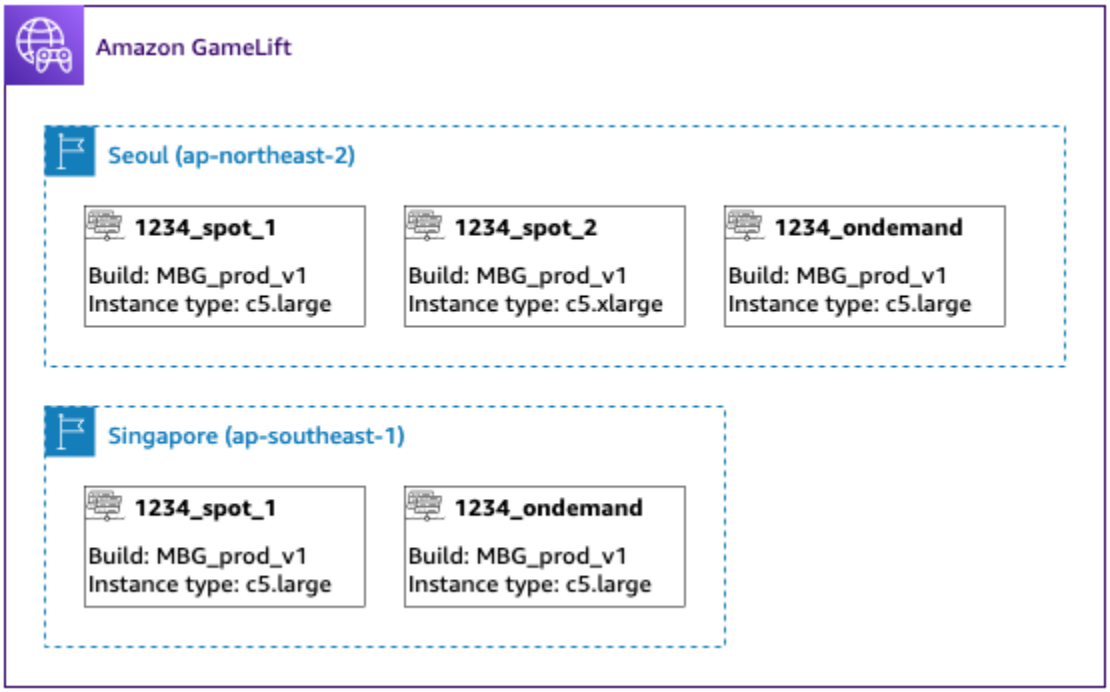
Para obtener más información, consulte Definición del ámbito de la cola.

Paso 2: Creación de una infraestructura de flota de spot

Cree flotas en ubicaciones y con compilaciones de servidores de juegos o scripts que se ajusten al ámbito definido en Paso 1: Definición del ámbito de la cola.

En este tutorial, crearemos una infraestructura de dos ubicaciones con al menos una flota de spot y una flota bajo demanda en cada ubicación. Todas las flotas implementan la misma compilación del servidor de juegos. Además, prevemos que el tráfico de jugadores será mayor en la ubicación de Seúl, por lo que añadiremos más flotas de spot allí.

En el siguiente diagrama se muestra un ejemplo de infraestructura de flota de spot, con 3 flotas en la ubicación ap-northeast-2 (Seúl) y 2 flotas en la ubicación ap-southeast-1 (Singapur). Todas las instancias de ambas flotas utilizan la compilación MBG_prod_V1. La flota de ap-northeast-2 contiene las siguientes configuraciones de flota: flota 1234_spot_1 con un tipo de instancia c5.large, flota 1234_spot_2 con un tipo de instancia c5.xlarge y flota 1234_ondemand con un tipo de instancia c5.large. La flota de ap-southeast-1 contiene las siguientes configuraciones de flota: flota 1234_spot_1 con un tipo de instancia de c5.large y flota 1234_ondemand con un tipo de instancia de c5.large.

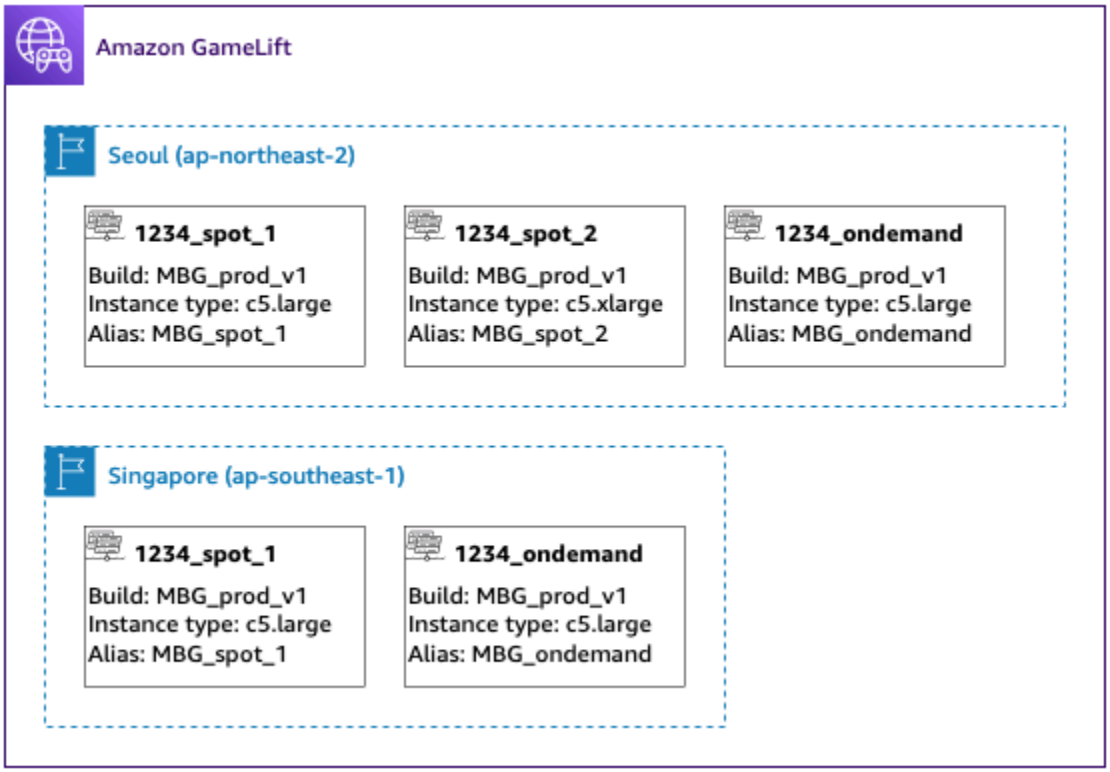


Paso 3: Asignación de alias a cada flota

Cree un nuevo alias para cada flota de su infraestructura. Los alias abstraen las identidades de la flota, lo que hace que el reemplazo periódico de la flota sea eficiente. Para obtener más información sobre la creación de alias, consulte Añadir un alias a una GameLift flota de Amazon.

Nuestra infraestructura de flota tiene cinco flotas, por lo que creamos cinco alias mediante la estrategia de enrutamiento. Necesitamos tres alias en la ubicación Asia-Pacífico (Seúl) y dos en la ubicación Asia-Pacífico (Singapur).

El siguiente diagrama muestra la infraestructura de flota de spot descrita en el paso dos con los alias añadidos a cada flota. La flota 1234_spot_1 tiene el alias MBG_spot_1, la flota 1234_spot_2 tiene el alias MBG_spot_2 y la flota 1234_ondemand tiene el alias MBG_ondemand.



Para obtener más información, consulte Creación una cola con varias ubicaciones.

Paso 4: Creación de una cola con destinos

Cree la cola de las sesiones del juego y añada los destinos de su flota. Para obtener más información sobre la creación de una cola, consulte Creación de una cola de sesión de juego.

Al crear la cola:

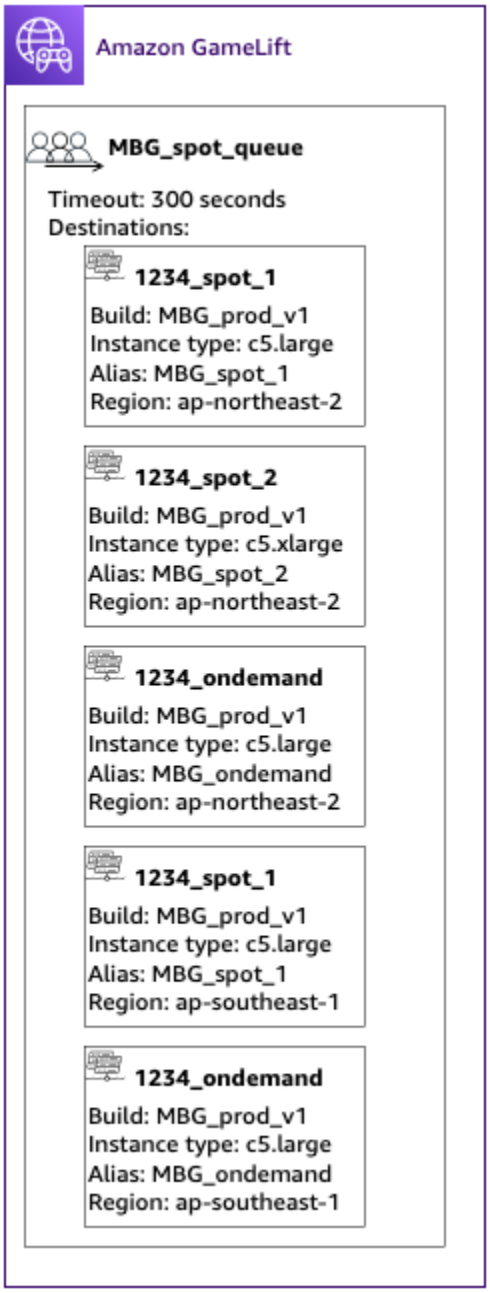
- Establezca el valor predeterminado del tiempo de espera en 10 minutos. Más adelante, podrá comprobar cómo afecta el tiempo de espera de la cola a los tiempos de espera de los jugadores para entrar en los juegos.
- Omita por ahora la sección sobre las políticas de latencia de los jugadores. Nos ocuparemos de ello en el siguiente paso.
- Priorice las flotas de la cola. Cuando trabaje con flotas de spot, le recomendamos que adopte uno de los siguientes enfoques:
  - Si su infraestructura utiliza una ubicación principal con las flotas en una segunda ubicación como respaldo, priorice las flotas primero por ubicación y luego por tipo de flota.

- Si su infraestructura utiliza varias ubicaciones por igual, priorice las flotas por tipo de flota y coloque las flotas de spot en la parte superior de la lista.

Para este tutorial, crearemos una nueva cola con el nombre **`MBG_spot_queue`** y añadiremos los alias de nuestras cinco flotas. Después, priorizaremos las ubicaciones primero por ubicación y, en segundo lugar, por tipo de flota.

Según esta configuración, esta lista siempre intentará incluir nuevas sesiones de juego en una flota de spot en Seúl. Cuando esas flotas estén llenas, la cola utilizará la capacidad disponible en la flota bajo demanda de Seúl como reserva. Si las tres flotas de Seúl no están disponibles, Amazon GameLift ubicará las sesiones de juego en las flotas de Singapur.

En el siguiente diagrama se muestra una cola con un tiempo de espera de 300 segundos y los destinos priorizados. Los destinos están en el siguiente orden: 1234_spot_1 en ap-northeast-2, 1234_spot_2 en ap-northeast-2, 1234_ondemand en ap-northeast-2, 1234_spot_1 en ap-southeast-1 y 1234_ondemand en ap-southeast-1.



Paso 5: Adición de límites de latencia a la cola

Nuestro juego incluye información de latencia en las solicitudes de ubicación de las sesiones de juego. También tenemos una característica de grupo de jugadores que crea una sesión de juego para un grupo de jugadores. Podemos hacer que los jugadores esperen un poco más para empezar a jugar con la experiencia de juego ideal. Nuestras pruebas de juego muestran las siguientes observaciones:

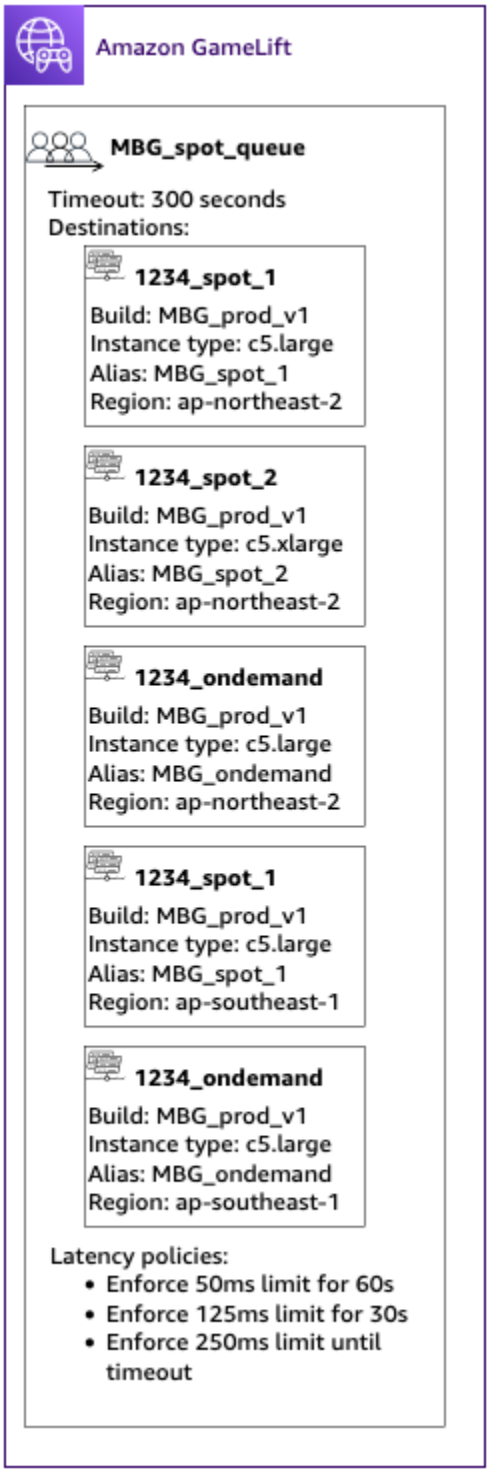
- Lo ideal es una latencia inferior a 50 milisegundos.

- El juego no se puede jugar con latencias superiores a 250 milisegundos.
- Los jugadores tardan aproximadamente un minuto en impacientarse.

Para nuestra cola, con un tiempo de espera de 300 segundos, añadimos instrucciones de política que limitan la latencia permitida. Las instrucciones de política permiten gradualmente valores de latencia más altos, de hasta 250 milisegundos.

Con esta política, nuestra cola busca ubicaciones con una latencia ideal (inferior a 50 milisegundos) durante el primer minuto y, a continuación, flexibiliza el límite. La cola no incluye ubicaciones en las que la latencia de los jugadores sea de 250 milisegundos o más.

El siguiente diagrama muestra la cola del paso cuatro con las políticas de latencia de los jugadores añadidas. Las políticas de latencia de los jugadores establecen el límite de 50 ms durante 60 segundos, el límite de 125 ms durante 30 segundos y el límite de 250 ms hasta que se agote el tiempo de espera.



Resumen

¡Enhorabuena! Estos son los objetivos conseguidos:

- Tiene una cola de sesiones de juego limitada a un segmento de la población de jugadores.

- La cola utiliza las flotas de spot de forma eficaz y es resiliente cuando se producen interrupciones de spot.
- La cola prioriza las flotas para que la experiencia de los mejores jugadores sea la mejor.
- La cola dispone de límites de latencia para proteger a los jugadores de malas experiencias de juego.

Ahora puede utilizar la cola para ubicar sesiones de juego para los jugadores a los que sirve. Al realizar solicitudes de ubicación de sesiones de juego para esos jugadores, hace referencia al nombre de la cola de sesiones de juego en la solicitud. Para obtener más información sobre cómo realizar solicitudes de ubicación en sesiones de juego, consulte Creación de sesiones de juego o Integración de un cliente de juegos para Realtime Servers.

Pasos siguientes:

- Diseño de su propia cola.
- Creación de una cola.
- Uso de una cola con el cliente de juegos.

Administración de recursos mediante AWS CloudFormation

Puede utilizar AWS CloudFormation para administrar sus recursos de Amazon GameLift. En AWS CloudFormation, puede crear una plantilla que sirva como modelo para cada recurso y, a continuación, utilizarla para crear los recursos. Para actualizar recursos, realice los cambios en la plantilla y utilice AWS CloudFormation para implementar las actualizaciones. Puede organizar sus recursos en grupos lógicos, llamados "pilas" y "conjuntos de pilas".

El uso de AWS CloudFormation para mantener sus recursos de alojamiento de Amazon GameLift ofrece una forma más eficiente de administrar conjuntos de recursos de AWS. Puede utilizar el control de versiones para realizar un seguimiento de los cambios en las plantillas a lo largo del tiempo y coordinar las actualizaciones realizadas por varios miembros del equipo. También puede reutilizar las plantillas. Por ejemplo, al implementar un juego en varias regiones, puede usar la misma plantilla para crear recursos idénticos en cada región. También puede utilizar estas plantillas para implementar los mismos conjuntos de recursos en otra partición.

Para obtener más información sobre AWS CloudFormation, consulte la AWS CloudFormationGuía del usuario de . Para ver información de la plantilla de recursos de Amazon GameLift, consulte la Referencia de tipos de recursos de Amazon GameLift.

Prácticas recomendadas

Para obtener instrucciones detalladas sobre el uso de AWS CloudFormation, consulte Prácticas recomendadas de AWS CloudFormation en la Guía del usuario de AWS CloudFormation. Además, estas prácticas recomendadas tienen especial relevancia cuando se utiliza Amazon GameLift.

- Administre de forma coherente los recursos que utiliza mediante AWS CloudFormation. Si cambia sus recursos fuera de AWS CloudFormation, sus recursos se desincronizarán de sus plantillas de recursos.
- Utilice pilas y conjuntos de pilas de AWS CloudFormation para administrar eficazmente varios recursos.
  - Utilice pilas para administrar grupos de recursos conectados. Por ejemplo, una pila que contiene una compilación, una flota que hace referencia a la compilación y un alias que hace referencia a la flota. Si actualiza la plantilla para reemplazar una compilación, AWS CloudFormation reemplaza las flotas conectadas a la compilación. AWS CloudFormation actualiza después los alias existentes para que apunten a las nuevas flotas. Para obtener más información, consulte Trabajo con pilas en la Guía del usuario de AWS CloudFormation.
  - Utilice conjuntos de pilas de AWS CloudFormation si va a implementar pilas idénticas en varias regiones o cuentas de AWS. Para obtener más información, consulte Trabajo con conjuntos de pilas en la Guía del usuario de AWS CloudFormation.
- Si utiliza instancias de spot, incluya una flota bajo demanda como respaldo. Es recomendable que configure las plantillas con dos flotas en cada región, una con instancias de spot y otra con instancias bajo demanda.
- Agrupe los recursos específicos de la ubicación y los recursos globales en pilas distintas cuando administre recursos en varias ubicaciones.
- Coloque sus recursos globales cerca de los servicios que los utilizan. Los recursos como las colas y las configuraciones de emparejamiento suelen recibir un gran volumen de solicitudes de orígenes específicos. Al colocar sus recursos cerca del origen de esas solicitudes, minimiza el tiempo de viaje de la solicitud y puede mejorar el rendimiento general.
- Coloque su configuración de emparejamiento en la misma región que la cola de sesiones del juego que utiliza.
- Cree un alias distinto para cada flota de la pila.

Uso de pilas de AWS CloudFormation

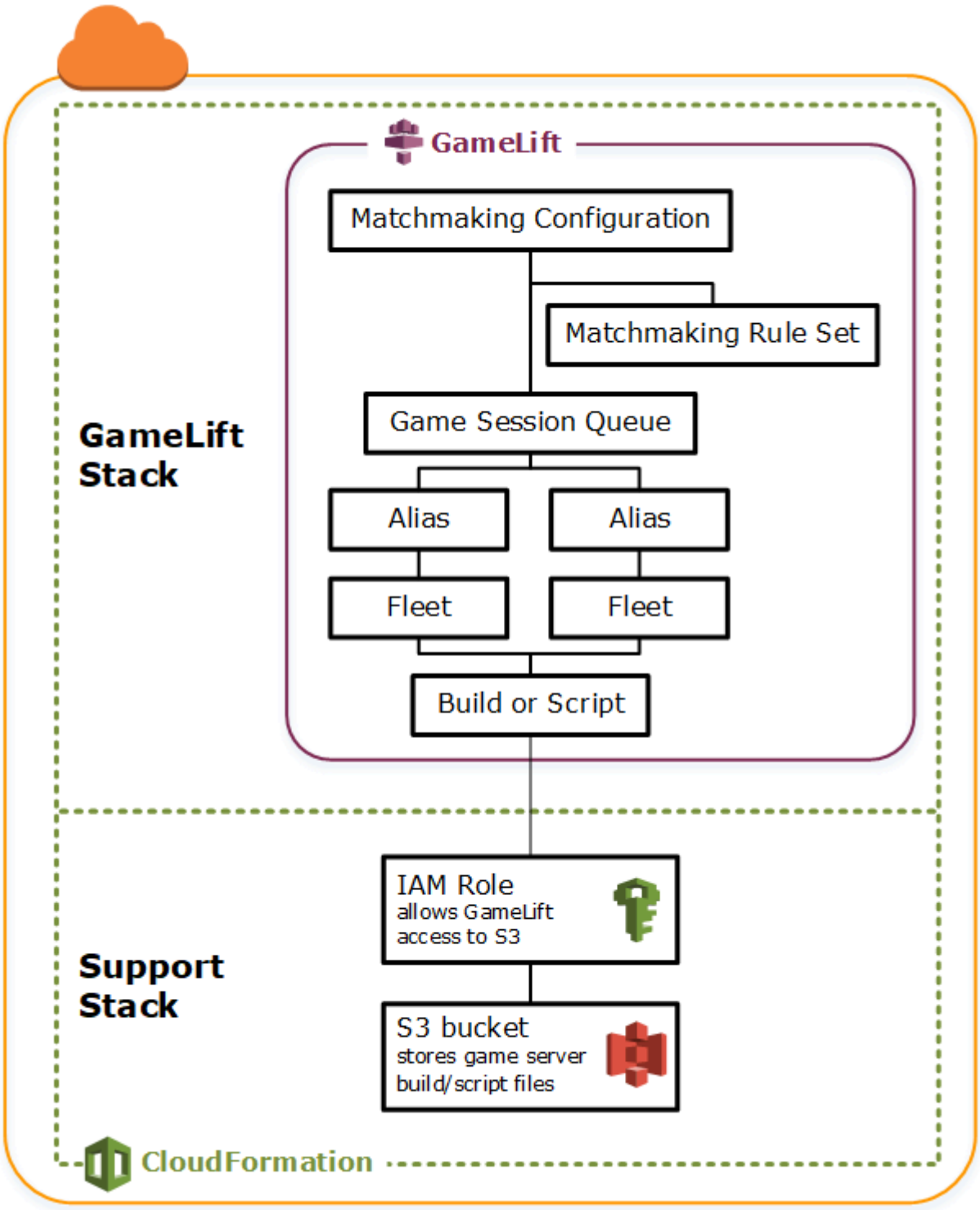
Recomendamos utilizar las siguientes estructuras recomendadas al configurar pilas de AWS CloudFormation para recursos de Amazon GameLift. La estructura óptima de la pila varía en función de si va a implementar el juego en una sola región o en varias regiones.

Pilas para una sola ubicación

Para administrar los recursos de Amazon GameLift en una sola ubicación, recomendamos una estructura de dos pilas:

- Pila de respaldo: esta pila contiene recursos de los que dependen los recursos de Amazon GameLift. Como mínimo, esta pila debe incluir el bucket de S3 donde almacena el servidor de juegos personalizado o los archivos de script de Realtime. La pila también debe incluir un rol de IAM que conceda a Amazon GameLift permiso para recuperar los archivos del bucket de S3 al crear un recurso de compilación o script de Amazon GameLift. Esta pila también puede contener otros recursos de que se utilicen con el juego, como tablas de DynamoDB, clústeres de Amazon Redshift y funciones de Lambda.
- Pila de Amazon GameLift: esta pila contiene todos los recursos de Amazon GameLift, incluida la compilación o el script, un conjunto de flotas, alias y una cola de sesiones de juego. AWS CloudFormation crea un recurso de compilación o script con los archivos almacenados en la ubicación del bucket de S3 e implementa la compilación o el script en uno o más recursos de la flota. Cada flota debe tener su alias correspondiente. La cola de sesiones del juego hace referencia a algunos o todos los alias de la flota. Si utiliza FlexMatch para el emparejamiento, esta pila también contiene una configuración de emparejamiento y un conjunto de reglas.

El siguiente diagrama ilustra una estructura de dos pilas para implementar recursos en una sola región de AWS.



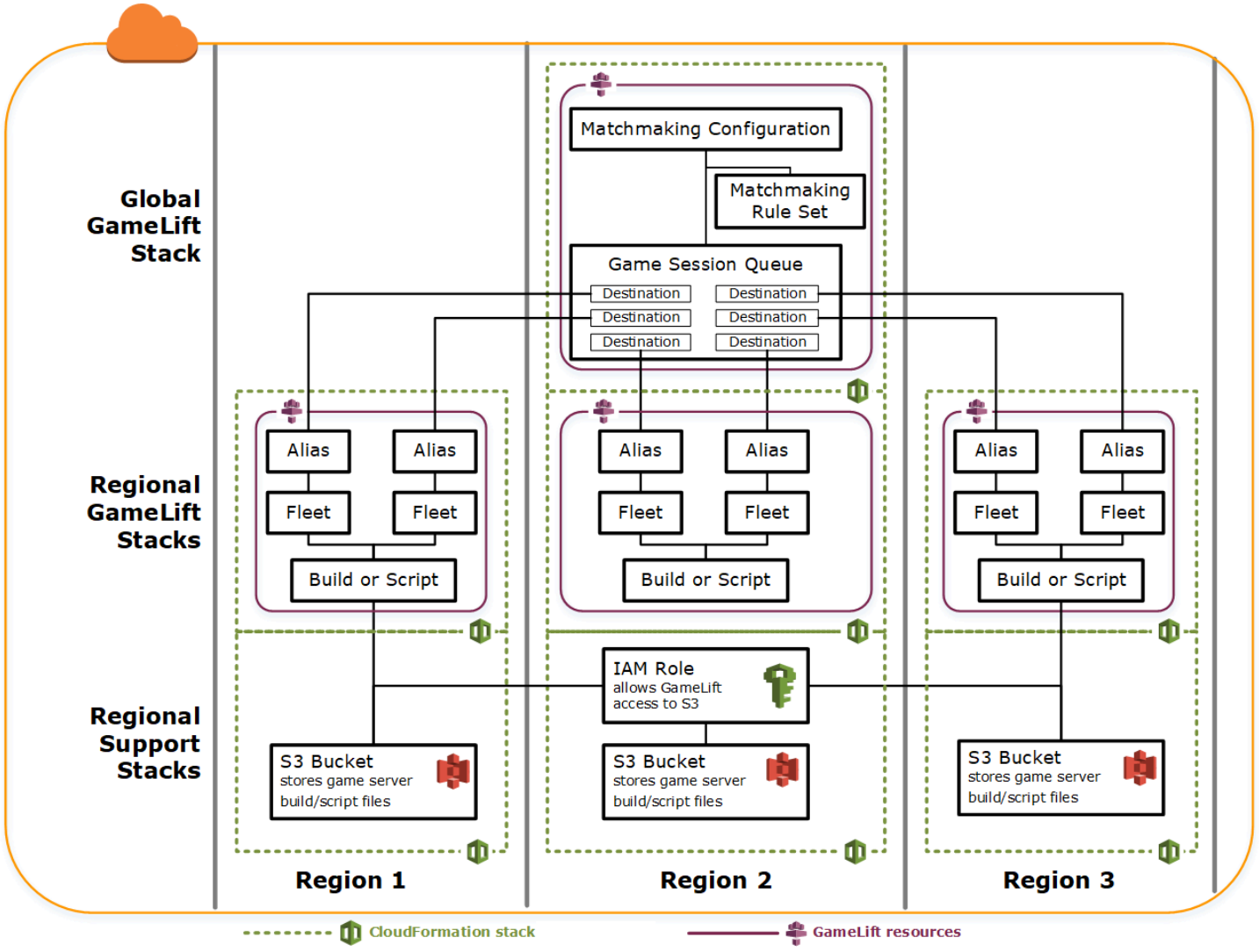
Pilas para varias regiones

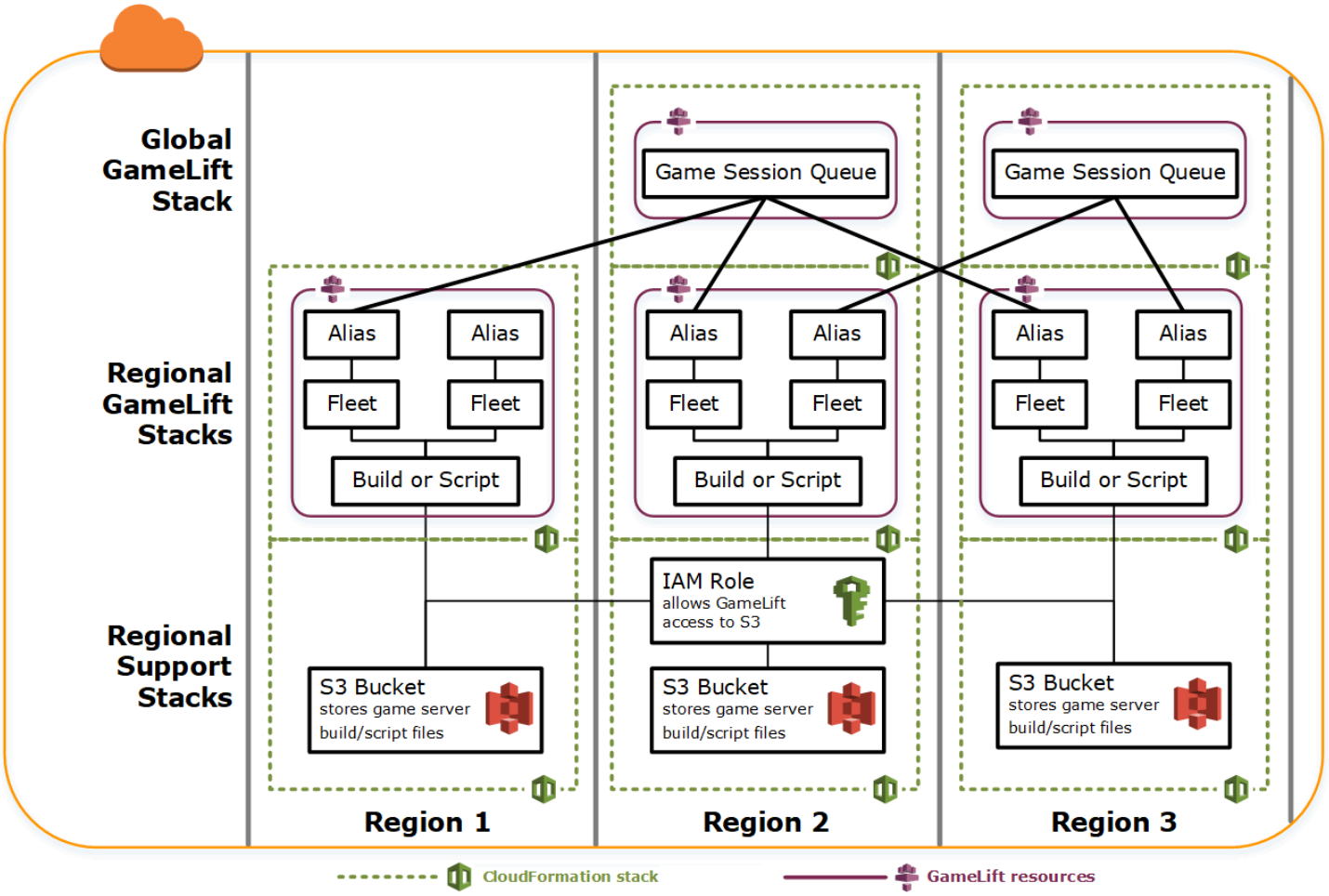
Al implementar el juego en más de una región, tenga en cuenta cómo los recursos pueden interactuar entre regiones. Algunos recursos, como las flotas de Amazon GameLift, solo pueden hacer referencia a otros recursos de la misma región. Otros recursos, como una cola de Amazon

GameLift, son independientes de la región. Para administrar recursos de Amazon GameLift en varias regiones, recomendamos la siguiente estructura.

- Pilas de respaldo regional: estas pilas contienen recursos de los que dependen los recursos de Amazon GameLift. Esta pila debe incluir el bucket de S3 en el que almacena el servidor de juegos personalizado o los archivos de script de Realtime. También pueden contener otros recursos de AWS para el juego, como tablas de DynamoDB, clústeres de Amazon Redshift y funciones de Lambda. Muchos de estos recursos son específicos de cada región, por lo que debe crearlos en cada una. Amazon GameLift también necesita un rol de IAM que permita obtener acceso a estos recursos de soporte. Puesto que un rol de IAM es independiente de la región, solo necesita un recurso de rol, colocado en cualquier región y al que se haga referencia en todas las demás pilas de respaldo.
- Pilas de Amazon GameLift regionales: esta pila contiene los recursos de Amazon GameLift, que debe haber en cada región en la que se implementa el juego, incluida la compilación o el script, un conjunto de flotas y alias. AWS CloudFormation crea un recurso de compilación o script con los archivos en una ubicación del bucket de S3 e implementa la compilación o el script en uno o más recursos de la flota. Cada flota debe tener su alias correspondiente. La cola de sesiones del juego hace referencia a algunos o todos los alias de la flota. Puede mantener una plantilla para describir este tipo de pila y utilizarla para crear conjuntos de recursos idénticos en cada región.
- Pila global de Amazon GameLift: esta pila contiene la cola de sesiones de juego y los recursos de emparejamiento. Estos recursos se pueden colocar en cualquier región, aunque se suelen colocar en la misma región. La cola puede hacer referencia a flotas o alias que se encuentran en cualquier región. Para colocar colas adicionales en diferentes regiones, cree pilas globales adicionales.

Los diagramas siguientes ilustran una estructura de varias pilas para implementar recursos en varias regiones de AWS. El primer diagrama muestra una estructura para una sola cola de sesiones del juego. El segundo diagrama muestra una estructura con varias colas.


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



Actualización de las compilaciones

Las compilaciones de Amazon GameLift son inmutables, al igual que la relación entre una compilación y una flota. Por consiguiente, cuando actualice sus recursos de alojamiento para usar un nuevo conjunto de archivos de compilación del juego, proceda de la siguiente forma:

- Cree una nueva compilación usando el nuevo conjunto de archivos (reemplazo).
- Cree un nuevo conjunto de flotas para implementar la nueva compilación del juego (reemplazo).
- Redirija los alias para que apunten a las nuevas flotas (actualización sin interrupción).

Para obtener más información, consulte Comportamientos de actualización de los recursos de la pila en la Guía del usuario de AWS CloudFormation.

Implementación de actualizaciones de compilación automáticamente

Al actualizar una pila que contiene recursos de compilación, flota y alias relacionados, el comportamiento predeterminado de AWS CloudFormation es realizar automáticamente estos pasos en orden. Esta actualización se activa al cargar primero los nuevos archivos de compilación en una nueva ubicación de S3. A continuación, modifica la plantilla de compilación de AWS CloudFormation para que apunte a la nueva ubicación de S3. Cuando actualiza la pila con la nueva ubicación S3, se desencadena la siguiente secuencia de AWS CloudFormation:

1. Se recuperan los nuevos archivos de S3, se validan los archivos y se crea una nueva compilación de Amazon GameLift.
2. Se actualiza la referencia de compilación en la plantilla de flota, lo que desencadena la creación de una nueva flota.
3. Una vez activadas las nuevas flotas, se actualiza la referencia de flota en el alias, lo que hace que el alias se actualice para apuntar a las nuevas flotas.
4. Se elimina la flota antigua.
5. Se elimina la compilación antigua.

Si la cola de sesiones del juego utiliza alias de flota, el tráfico de jugadores se cambia automáticamente a las nuevas flotas en cuanto se actualizan los alias. Las flotas antiguas se van quedando poco a poco sin jugadores a medida que terminan las sesiones del juego. El escalado automático controla la tarea de añadir y eliminar instancias de cada conjunto de flotas a medida que fluctúa el tráfico de jugadores. También puede especificar un recuento inicial de las instancias deseadas para que el cambio se produzca rápidamente y habilitar el escalado automático más adelante.

También puede mantener los recursos de AWS CloudFormation en lugar de eliminarlos. Para obtener más información, consulte RetainResources en la Referencia de la API de AWS CloudFormation.

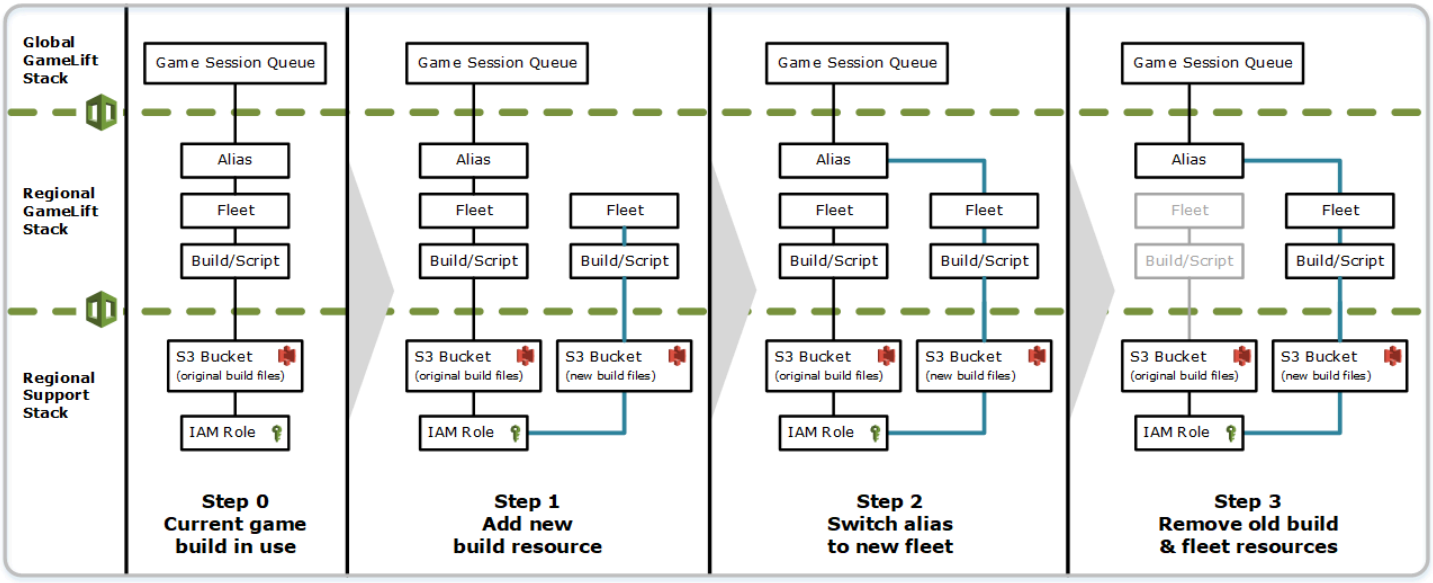
Implementación de actualizaciones de compilación manualmente

Si desea tener más control sobre cuándo se activan las nuevas flotas para los jugadores, dispone de algunas opciones. Puede elegir administrar los alias manualmente mediante la consola de Amazon GameLift o la CLI. Otra opción es añadir un segundo conjunto de definiciones de compilación y flota en la plantilla, en lugar de actualizar la plantilla de compilación para reemplazar la compilación y las flotas. Al actualizar la plantilla, AWS CloudFormation crea un segundo recurso de compilación y las

flotas correspondientes. Dado que los recursos existentes no se reemplazan, estos no se eliminan y los alias siguen apuntando a las flotas originales.

La principal ventaja de este enfoque es que le ofrece flexibilidad. Puede crear recursos distintos para la nueva versión de su compilación, probar los nuevos recursos y controlar cuándo se activan las nuevas flotas para los jugadores. Un posible inconveniente es que requiere el doble de recursos en cada región durante un breve período de tiempo.

En el siguiente diagrama se ilustra este proceso.



Cómo funcionan las restauraciones

Al ejecutar una actualización de recursos, si algún paso no se completa correctamente, AWS CloudFormation inicia automáticamente una restauración. Este proceso invierte cada paso en orden, eliminando los recursos recién creados.

Si necesita activar manualmente una restauración, cambie la clave de ubicación de S3 de la plantilla de compilación a la ubicación original y actualice la pila. Se crea una nueva compilación y flota de Amazon GameLift, y el alias cambia a la nueva flota una vez que la flota esté activa. Si administra alias por separado, debe cambiarlos para que apunten a las nuevas flotas.

Para obtener más información sobre cómo controlar una restauración que da un error o se bloquea, consulte Continuación de la restauración de una actualización en la Guía del usuario de AWS CloudFormation.

Emparejamiento de VPC para Amazon GameLift

En este tema, se ofrecen directrices sobre cómo configurar una conexión de emparejamiento de VPC entre sus servidores de juegos alojados en Amazon GameLift y otros recursos que no son de Amazon GameLift. Utilice las conexiones de emparejamiento de la nube privada virtual (VPC) de Amazon para permitir que sus servidores de juegos se comuniquen directamente y de forma privada con otros de sus recursos de AWS, como un servicio web o un repositorio. Puede establecer un emparejamiento de VPC con cualquier recurso que se ejecute en AWS y se administre mediante una cuenta de AWS a la que tenga acceso.

> **Note**
>
> La interconexión de VPC es una característica avanzada. Si desea conocer las opciones preferidas para permitir que sus servidores de juegos se comuniquen directamente y de forma privada con sus otros recursos de AWS, consulte Comunicación con otros recursos de AWS de sus flotas.

Si ya está familiarizado con las VPC y el emparejamiento de VPC de Amazon, tenga en cuenta que la configuración del emparejamiento con los servidores de juegos de Amazon GameLift es algo distinta. No dispone de acceso a la VPC que contiene los servidores de juegos (que se controla mediante el servicio de Amazon GameLift), por lo que no puede solicitar directamente el emparejamiento de la VPC. En lugar de ello, primero debe preautorizar la VPC con recursos que no pertenecen a Amazon GameLift para aceptar una solicitud de emparejamiento del servicio de Amazon GameLift. A continuación, debe hacer que Amazon GameLift solicite el emparejamiento de la VPC que acaba de autorizar. Amazon GameLift se encarga de las tareas de crear la conexión de emparejamiento, configurar las tablas de enrutamiento y configurar la conexión.

Para configurar la interconexión de VPC para una flota existente

1. Obtenga los ID de cuenta de AWS y las credenciales.

   Necesita un ID y credenciales de inicio de sesión para las siguientes cuentas de AWS. Inicie sesión en la AWS Management Console y consulte la configuración de su cuenta para encontrar los ID de cuenta de AWS. Para obtener las credenciales, vaya a la consola de IAM.

   - Cuenta de AWS que se utiliza para administrar los servidores de juegos de Amazon GameLift.

- Cuenta de AWS que se utiliza para administrar los recursos que no pertenecen a Amazon GameLift.

Si utiliza la misma cuenta para recursos de Amazon GameLift y otros que no sean de Amazon GameLift, necesita solo el ID y las credenciales de esa cuenta.

2. Obtenga los identificadores de cada VPC.

   Obtenga la siguiente información para las dos VPC que se van a interconectar:

   - VPC para sus servidores de juegos de Amazon GameLift:, es decir, el ID de la flota de Amazon GameLift. Los servidores de juegos están implementados en Amazon GameLift en una flota de instancias EC2. Una flota se sitúa automáticamente en su propia VPC, que se administra mediante el servicio de Amazon GameLift. No dispone de acceso directo a la VPC, por lo que se identifica mediante el ID de la flota.
   - VPC para los recursos de AWS que no pertenecen a Amazon GameLift: puede establecer una conexión de emparejamiento de VPC con cualquier recurso que se ejecute en Amazon GameLift y se administre mediante una cuenta de AWS a la que tenga acceso. Si todavía no ha creado una VPC para esos recursos, consulte Introducción a Amazon VPC. Una vez que haya creado una VPC, puede encontrar el ID de esta iniciando sesión en la AWS Management Console de Amazon VPC y consultando las VPC.

   > **Note**
   >
   > Cuando se configura una interconexión, las dos VPC deben existir en la misma región. La VPC de los servidores de juegos de la flota de Amazon GameLift está en la misma región que la flota.

3. Autorizar una interconexión de VPC.

   En este paso, preautorizará una solicitud futura de Amazon GameLift para conectar la VPC con los servidores de juegos con la VPC para recursos que no pertenecen a Amazon GameLift. Esta acción actualiza el grupo de seguridad de la VPC.

   Para autorizar el emparejamiento de la VPC, llame a la API de servicio de Amazon GameLift CreateVpcPeeringAuthorization() o utilice el comando de la CLI de AWS `create-vpc-peering-authorization`. Realice esta llamada mediante la cuenta que administra los recursos que no pertenecen a Amazon GameLift. Proporcione la siguiente información:

- ID de VPC de par: es para la VPC con los recursos que no son de Amazon GameLift.
- ID de cuenta de AWS de Amazon GameLift: es la cuenta que utiliza para administrar la flota de Amazon GameLift.

Una vez que haya autorizado la interconexión de VPC, la autorización sigue siendo válida durante 24 horas, a menos que revoque. Puede administrar las autorizaciones de interconexión de VPC mediante las siguientes operaciones:

- DescribeVpcPeeringAuthorizations() (`describe-vpc-peering-authorizations` de la CLI de AWS).
- DeleteVpcPeeringAuthorization() (`delete-vpc-peering-authorization` de la CLI de AWS).

4. Solicite una interconexión.

   Con una autorización válida, puede solicitar que Amazon GameLift establezca un emparejamiento.

   Para solicitar el emparejamiento de la VPC, llame a la API de servicio de Amazon GameLift CreateVpcPeeringConnection() o utilice el comando `create-vpc-peering-connection` de la CLI de AWS. Realice esta llamada con la cuenta que administra los servidores de juego de Amazon GameLift. Utilice la siguiente información para identificar las dos VPC que desea interconectar:

   - ID de la VPC de par e ID de la cuenta de AWS: es la VPC de los recursos que no pertenecen a Amazon GameLift y la cuenta que utiliza para la administrarlos. El ID de la VPC debe coincidir con el ID de una autorización de interconexión válida.
   - ID de la flota: identifica la VPC de los servidores de juegos de Amazon GameLift.

5. Realice un seguimiento del estado de la interconexión.

   La solicitud de una interconexión de VPC es una operación asíncrona. Para realizar el seguimiento del estado de una solicitud de interconexión y gestionar los casos de éxito o error, utilice una de las siguientes opciones:

   - Sondee continuamente con `DescribeVpcPeeringConnections()`. Esta operación recupera el registro de la interconexión de VPC, incluido el estado de la solicitud. Si se crea

correctamente una interconexión, el registro de conexión también contiene un bloque de CIDR de direcciones IP privadas que se asigna a la VPC.

- Gestione los eventos de flota asociados a las inconexiones de VPC con DescribeFleetEvents(), incluidos los eventos de éxito y de error.

Una vez establecida la interconexión, puede administrarla mediante las siguientes operaciones:

- DescribeVpcPeeringConnections() (`describe-vpc-peering-connections` de la CLI de AWS).
- DeleteVpcPeeringConnection() (`delete-vpc-peering-connection` de la CLI de AWS).

Para configurar la interconexión de VPC con una nueva flota

Puede crear una nueva flota de Amazon GameLift y solicitar una conexión de emparejamiento de VPC al mismo tiempo.

1. Obtenga los ID de cuenta de AWS y las credenciales.

   Necesita un ID y credenciales de inicio de sesión para las siguientes dos cuentas de AWS. Inicie sesión en la AWS Management Console y consulte la configuración de su cuenta para encontrar los ID de cuenta de AWS. Para obtener las credenciales, vaya a la consola de IAM.

   - Cuenta de AWS que se utiliza para administrar los servidores de juegos de Amazon GameLift.
   - Cuenta de AWS que se utiliza para administrar los recursos que no pertenecen a Amazon GameLift.

   Si utiliza la misma cuenta para recursos de Amazon GameLift y otros que no sean de Amazon GameLift, necesita solo el ID y las credenciales de esa cuenta.
2. Obtenga el ID de VPC de sus recursos de AWS que no pertenecen a Amazon GameLift.

   Si todavía no ha creado una VPC para estos recursos, hágalo ahora (consulte Introducción a Amazon VPC). Asegúrese de crear la nueva VPC en la misma región en la que piensa crear su nueva flota. Si los recursos que no son de Amazon GameLift se administran en una cuenta de AWS o usuario/grupo de usuarios distintos de los que utiliza con Amazon GameLift, deberá utilizar las credenciales de esa cuenta al solicitar la autorización en el siguiente paso.

   Una vez que haya creado una VPC, puede localizar el ID de la VPC en la consola de Amazon VPC consultando las VPC.

3. Autorice un emparejamiento de la VPC con los recursos que no son de Amazon GameLift.

   Cuando Amazon GameLift crea la nueva flota y la VPC correspondiente, también envía una solicitud para emparejar la VPC para los recursos que no pertenecen a Amazon GameLift. Debe para preautorizar dicha solicitud. Este paso actualiza el grupo de seguridad de la VPC.

   Utilice las credenciales de la cuenta que administra los recursos que no son de Amazon GameLift, llame a la API de servicio de Amazon GameLift CreateVpcPeeringAuthorization() o utilice el comando de la CLI de AWS `create-vpc-peering-authorization`. Proporcione la siguiente información:

   - ID de VPC de par: ID de la VPC con los recursos que no pertenecen a Amazon GameLift.
   - ID de cuenta de AWS de Amazon GameLift: ID de la cuenta que utiliza para administrar la flota de Amazon GameLift.

   Una vez que haya autorizado la interconexión de VPC, la autorización sigue siendo válida durante 24 horas, a menos que se revoque. Puede administrar las autorizaciones de interconexión de VPC mediante las siguientes operaciones:

   - DescribeVpcPeeringAuthorizations() (`describe-vpc-peering-authorizations` de la CLI de AWS).
   - DeleteVpcPeeringAuthorization() (`delete-vpc-peering-authorization` de la CLI de AWS).
4. Siga las instrucciones para crear una nueva flota a través de la CLI de AWS. Incluya los siguientes parámetros adicionales:

   - peer-vpc-aws-account-id: ID para la cuenta que utiliza para administrar la VPC con sus recursos que no son de Amazon GameLift.
   - peer-vpc-id: ID de la VPC con la cuenta que no pertenece a GameLift.

Una llamada correcta a create-fleet con los parámetros de la interconexión de VPC genera una flota nueva y una nueva solicitud de interconexión de VPC. El estado de la flota se establece en New y se inicia el proceso de activación de la flota. El estado de la solicitud de interconexión se establece en initiating-request. Puede comprobar si la solicitud de interconexión se realiza o no correctamente llamando a describe-vpc-peering-connections.

Cuando se solicita una nueva flota y una interconexión de VPC, las dos acciones se realizarán correctamente o producirán un error. Si se produce un error en una flota durante el proceso de creación, la interconexión de VPC no se establecerá. Del mismo modo, si falla una interconexión de VPC por cualquier motivo, la flota nueva no podrá a pasar del estado Activating al estado Active.

> **Note**
>
> La nueva interconexión de VPC no se completará hasta que la flota esté lista para activarse. Esto significa que la conexión no está disponible y no se puede usar durante el proceso de instalación de la compilación del servidor de juegos.

En el siguiente ejemplo se crea una nueva flota y una interconexión entre una VPC previamente establecida y la VPC de la nueva flota. La VPC preestablecida se identifica de forma inequívoca al combinar el ID de la cuenta de AWS que no es de Amazon GameLift y el ID de la 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"
```

Versión 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"
```

Solución de problemas de interconexión de VPC

Si tiene problemas para establecer un emparejamiento de la VPC para sus servidores de juegos de Amazon GameLift, tenga en cuenta estas causas principales comunes:

- No se ha encontrado una autorización para la conexión solicitada:
  - Compruebe el estado de una autorización de la VPC para la VPC que no es de Amazon GameLift. Es posible que no exista o que haya caducado.
  - Compruebe las regiones de las dos VPC que intenta interconectar. Si no están en la misma región, no se pueden interconectar.
- Los bloques de CIDR (consulte Configuraciones del emparejamiento de la VPC no válidas) de las dos VPC se superponen. Los bloques de CIDR IPv4 asignados a las VPC emparejadas no pueden solaparse. El bloque de CIDR de la VPC de su flota de Amazon GameLift se asigna automáticamente y no se puede cambiar, por lo que tendrá que cambiar el bloque de CIDR de la VPC para los recursos no pertenecientes a Amazon GameLift. Para resolver este problema, siga estos pasos:
  - Busque este bloque de CIDR para su flota de Amazon GameLift llamando a `DescribeVpcPeeringConnections()`.
  - Vaya a la consola de Amazon VPC, busque la VPC para los recursos que no son de Amazon GameLift y cambie el bloque de CIDR para que no se superpongan.
- La nueva flota no se activó (al solicitar la interconexión de VPC con una nueva flota). Si la nueva flota no pudo avanzar hasta el estado Activo no hay ninguna VPC con la que interconectar, por lo que no se puede realizar la interconexión.

Visualización de los datos de juego en la consola

El servicio de Amazon GameLift administrado recopila datos continuamente para que los juegos activos le ayuden a comprender el comportamiento y el rendimiento de los jugadores. Puede utilizar la consola de Amazon GameLift para ver, administrar y analizar dicha información para las compilaciones, flotas y sesiones de juego y de jugador.

Temas

Ver tu GameLift estado actual en Amazon

El GameLift panel de control de Amazon ofrece una vista de lo siguiente:

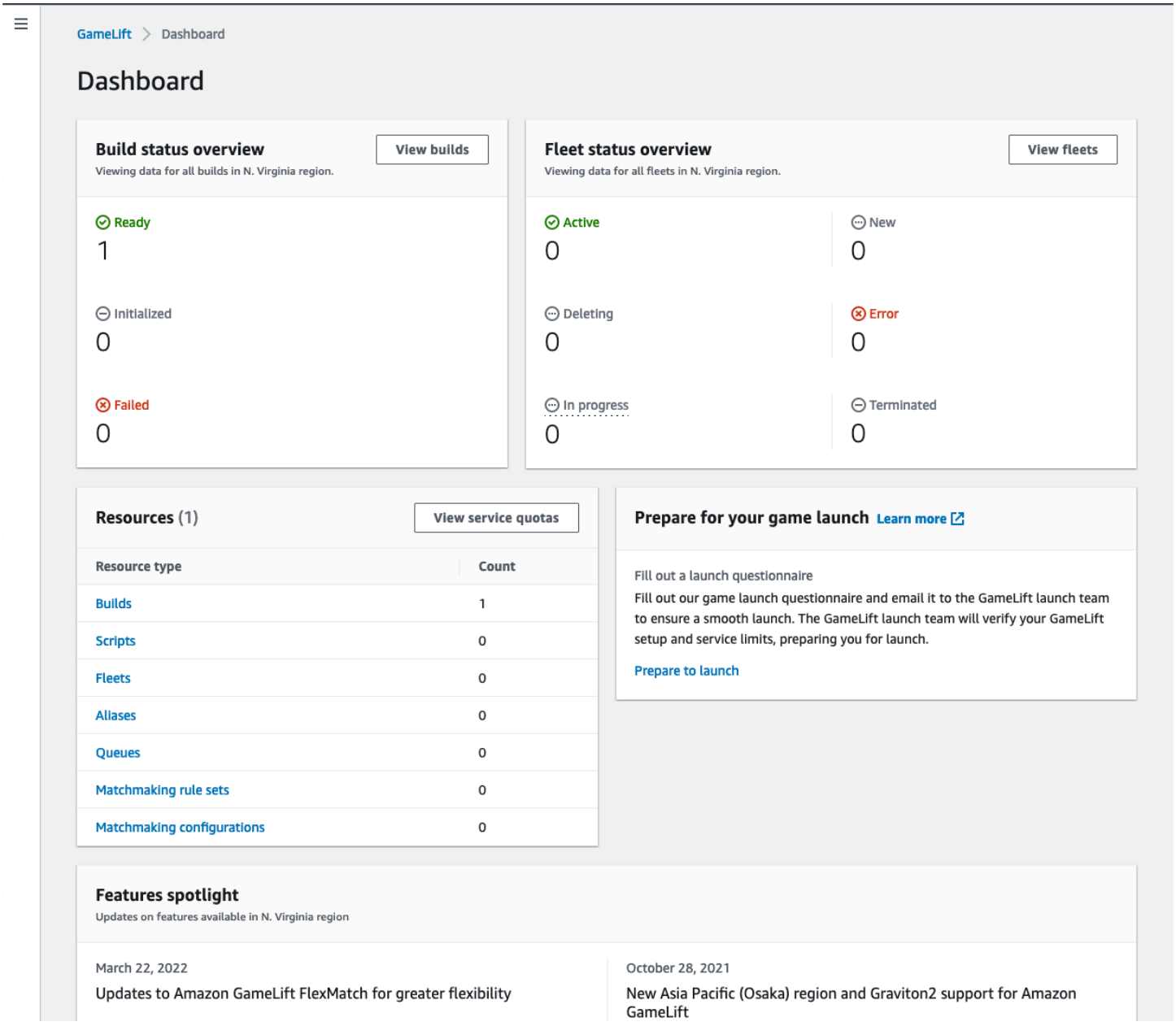
- El número de compilaciones con el estado Listo, Iniciado y Error. Elija Ver compilaciones para obtener más información sobre las compilaciones de su región actual.
- El número de flotas en todos los estados. Elija Ver compilaciones para obtener información sobre las flotas de su región actual.
- Sus recursos actuales.
- Anuncios de nuevas características y servicios.

Para abrir el GameLift panel de Amazon

- En la GameLift consola de Amazon, en el panel de navegación, selecciona Dashboard.

En el panel, podrá realizar los siguientes procedimientos:

- Para preparar el juego para el lanzamiento, seleccione Prepararse para el lanzamiento y complete el cuestionario de lanzamiento correspondiente.
- Para solicitar aumentos de cuota de servicio como preparación para los lanzamientos o como respuesta a los lanzamientos, elija Ver service quotas.
- Para ver las publicaciones del blog y la información detallada sobre las nuevas características, elija el enlace que aparece en Características destacadas.



Visualización de las compilaciones

En la página Compilaciones de la consola de Amazon GameLift podrá ver información sobre todas las compilaciones de servidores de juegos que ha cargado en Amazon GameLift y administrarlas. En el panel de navegación, elija Alojamiento, Compilaciones.

La página Compilaciones muestra la siguiente información para cada compilación:

> **Note**
>
> La página Compilaciones muestra solo las compilaciones de su región de AWS actual.

- Nombre: nombre asociado a la compilación cargada.
- Estado: estado de la compilación. Muestra uno de los tres mensajes de estado:
  - Iniciado: la carga no se ha iniciado o aún está en curso.
  - Preparado: la compilación está lista para la creación de la flota.
  - Error: la compilación ha expirado antes de que Amazon GameLift recibiera los archivos binarios.
- Hora de creación: fecha y hora en que cargó la implementación en Amazon GameLift.
- ID de compilación: ID único asignado al cargar la compilación.
- Versión: etiqueta de versión asociada a la compilación cargada.
- Sistema operativo: SO en el que se ejecuta la compilación. El SO de la compilación determina el sistema operativo que Amazon GameLift instala en las instancias de una flota.
- Tamaño: tamaño, en megabytes (MB), del archivo de compilación cargado en Amazon GameLift.
- Flotas: cantidad de flotas implementadas con la compilación.

En esta página puede hacer lo siguiente:

- Ver los detalles de las compilaciones. Elija el nombre de una compilación para abrir la página de detalles de la compilación.
- Crear una flota nueva a partir de una compilación. Seleccione una compilación y, a continuación, elija Crear flota.
- Filtrar y ordenar la lista de compilación. Utilice los controles en la parte superior de la tabla.
- Eliminar una compilación. Seleccione una compilación y, a continuación, elija Eliminar.

Detalles de la compilación

En la página Compilaciones, elija el nombre de una compilación para abrir la página de detalles sobre ella. La sección Información general de la página de detalles muestra la misma información resumida de la compilación que aparece en la página Compilaciones. La sección Flotas muestra una lista de las flotas creadas con la compilación, que incluye la misma información resumida que en la página Flotas.

Visualización de los scripts

En la página Scripts de la consola de Amazon GameLift podrá ver información sobre todos los scripts de Realtime Servers que ha cargado en Amazon GameLift y administrarlos. En el panel de navegación, elija Alojamiento, Scripts.

La página Scripts muestra la siguiente información para cada script:

> **Note**
>
> La página Scripts muestra solo los scripts solo en su región de AWS actual.

- Nombre: nombre asociado al script cargado.
- ID: ID único asignado al cargar el script.
- Versión: etiqueta de versión asociada al script cargado.
- Tamaño: tamaño, en megabytes (MB), del archivo de script cargado en Amazon GameLift.
- Hora de creación: fecha y hora en que cargó el script en Amazon GameLift.
- Flotas: cantidad de flotas implementadas con el script.

En esta página puede hacer lo siguiente:

- Visualizar los detalles del script. Elija el nombre de una compilación para abrir la página de detalles del script.
- Crear una flota nueva a partir del script. Seleccione un script y, a continuación, elija Crear flota.
- Filtrar y ordenar la lista de script. Utilice los controles en la parte superior de la tabla.
- Eliminar un script. Seleccione un script y, a continuación, elija Eliminar.

Detalles del script

En la página Scripts, elija el nombre de un script para abrir la página de detalles sobre él. La sección Información general de la página de detalles muestra la misma información resumida del script que aparece en la página Compilaciones. La sección Flotas muestra una lista de las flotas creadas con el script, que incluye la misma información resumida que en la página Flotas.

Visualización de flotas

Puede visualizar información sobre todas las flotas creadas para alojar los juegos en Amazon GameLift en su cuenta de AWS. La lista muestra las flotas creadas en su región actual. En la página Flotas, puede crear una flota nueva o ver información detallada adicional sobre una flota. La página de detalles de una flota contiene información de uso, métricas, datos de sesión de juego y datos de sesión de jugador. También puede editar el registro de una flota o eliminarla.

Para ver la página Flotas, elija Flotas en el panel de navegación.

De forma predeterminada, la página Fleets (Flotas) muestra la siguiente información resumida. Puede pulsar el botón Configuración (engranaje) para personalizar la información que aparece.

- Nombre: nombre fácil de recordar de la flota.
- Estado: estado actual de la flota, que puede tener uno de los siguientes estados Nuevo, Descargando, Creando y Activo.
- Hora de creación: la fecha y la hora en que se creó la flota.
- Tipo de computación: el tipo de computación utilizado para alojar sus juegos. Una flota puede ser una flota de EC2 administrada o una flota de Anywhere.
- Tipo de instancia: tipo de instancia de Amazon EC2 que determina la capacidad de procesamiento de las instancias de la flota.
- Instancias activas: número de instancias EC2 en uso para la flota.
- Instancias deseadas: cantidad de instancias de EC2 que se deben mantener activas.
- Sesiones de juego: número de sesiones de juego activas que se ejecutan en la flota. Los datos tienen una demora de cinco minutos.

Visualización de los detalles de la flota

Elija el nombre de la flota para acceder a la página de detalles Flota desde el panel o desde la página Flotas.

En la página Detalles de la flota puede llevar a cabo las siguientes acciones:

- Actualizar los atributos, los ajustes de los puertos y la configuración del tiempo de ejecución de una flota.
- Añadir o eliminar ubicaciones de flota.
- Cambiar la configuración de la capacidad de la flota.
- Establecer o cambiar el escalado automático del seguimiento de objetivos.
- Eliminar una flota.

Detalles

Configuración de la flota

- ID de la flota: identificador único asignado a la flota.
- Nombre: nombre de la flota.
- ARN: identificador asignado a esta flota. El ARN de una flota la identifica como un recurso de Amazon GameLift y especifica la región y la cuenta de AWS.
- Descripción: breve descripción identificable de la flota.
- Estado: estado actual de la flota, que puede ser Nuevo, Descargando, Creando y Activo.
- Hora de creación: la fecha y la hora en que se creó la flota.
- Hora de terminación: la fecha y la hora en que se canceló la flota. Está en blanco si la flota sigue activa.
- Tipo de flota: indica si la flota utiliza instancias de spot o bajo demanda.
- Tipo EC: tipo de instancia de Amazon EC2 seleccionado para la flota al crearla.
- Rol de instancia: rol de IAM de AWS que administra el acceso a otros recursos de AWS, si se proporcionó uno durante la creación de la flota.
- Certificado TLS: si la flota está habilitada o deshabilitada, puede utilizar un certificado TLS para autenticar un servidor de juegos y cifrar todas las comunicaciones entre el cliente y el servidor.
- Grupo de métricas: grupo de métricas utilizado para añadir las métricas de varias flotas.

- Política de protección para el escalado de juegos: configuración actual de la protección de la sesión de juego para la flota.
- Número máximo de sesiones de juego por jugador: número máximo de sesiones que un jugador puede crear durante el periodo de política.
- Periodo de política: tiempo que se debe esperar hasta que se restablezca el número de sesiones que ha creado un jugador.

Detalles de la compilación

La sección Detalles de la compilación muestra la compilación alojada en la flota. Seleccione el nombre de compilación para ver la página de detalles de la compilación completa.

Configuración del tiempo de ejecución

La sección Configuración del tiempo de ejecución muestra los procesos del servidor que se van a lanzar en cada instancia. Incluye la ruta del ejecutable del servidor de juegos y los parámetros de lanzamiento opcionales.

Activación de la sesión de juego

La sección Activación de la sesión de juego muestra el número de procesos del servidor que se inician al mismo tiempo y cuánto tiempo hay que esperar a que el proceso se active antes de finalizarlo.

Configuración de puertos de EC2

La sección Puertos muestra los permisos de conexión de la flota, incluida la dirección IP y los rangos de configuración del puerto.

Métricas

La pestaña Metrics (Métricas) muestra una representación gráfica de las métricas de la flota a lo largo del tiempo. Para obtener más información sobre el uso de métricas en Amazon GameLift, consulte Supervisión de Amazon GameLift con Amazon CloudWatch.

Eventos

La pestaña Events proporciona un registro de todos los eventos que se han producido en la flota, incluido el código del evento, el mensaje y la marca temporal. Consulte las descripciones de Evento en la Referencia de la API de Amazon GameLift

Escalado

La pestaña Escalado contiene información sobre la capacidad de la flota, incluido el estado actual y los cambios de capacidad a lo largo del tiempo. También proporciona herramientas para actualizar los límites de capacidad y gestionar el escalado automático.

Escalado de capacidad

Consulte la configuración de la capacidad de la flota actual para cada ubicación de la flota. Para obtener más información sobre el cambio de límites y de capacidad, consulte Escalación de la capacidad de alojamiento de Amazon GameLift.

- Ubicación de AWS: nombre de la ubicación en la que se implementan las instancias de flota.
- Estado: estado del alojamiento de la ubicación de la flota. El estado de la ubicación debe ser `ACTIVE` para poder alojar juegos.
- Tamaño mínimo: número mínimo de instancias que se deben implementar en la ubicación.
- Instancias deseadas: cantidad de destino de instancias activas para mantener la ubicación. Cuando las instancias activas y deseadas no son las mismas, se inicia un evento de escalado para iniciar o cerrar las instancias según sea necesario hasta que las instancias activas equivalgan a las instancias deseadas.
- Tamaño máximo: el mayor número de instancias que se puede implementar en la ubicación.
- Disponible: el límite de servicio de las instancias menos la cantidad de instancias en uso. Este valor indica el número máximo de instancias que puede añadir a la ubicación.

Políticas de escalado automático

En esta sección se incluye información sobre las políticas de escalado automático que se aplican a la flota. Puede configurar o actualizar una política basada en objetivos. Aquí se muestran las políticas basadas en reglas de la flota, que deben definirse mediante la CLI o el SDK de AWS. Para obtener más información sobre el escalado, consulte Escalado automático de la capacidad con Amazon GameLift.

Historial de escalado

Consulte los gráficos de los cambios de capacidad a lo largo del tiempo.

Locations

La pestaña Ubicaciones muestra todas las ubicaciones en las que se implementan las instancias de flota. Las ubicaciones incluyen la región de origen de la flota y cualquier ubicación remota que se haya añadido. Puede añadir o eliminar ubicaciones directamente en esta pestaña.

- Ubicación: nombre de la ubicación en la que se implementan las instancias de flota.
- Estado: estado del alojamiento de la ubicación de la flota. Con la opción Estado de la ubicación, se realiza un seguimiento del proceso de activación de las primeras instancias de la ubicación. Además, la opción debe tener el estado `ACTIVE` para poder alojar juegos.
- Instancias activas: cantidad de instancias con procesos de servidor que se ejecutan en la ubicación de la flota.
- Servidores activos: número de procesos del servidor de juegos que pueden alojar sesiones de juego en la ubicación de la flota.
- Sesiones de juego: número de sesiones de juego activas en instancias en la ubicación de la flota.
- Sesiones de jugador: número de sesiones de jugador, que representan a jugadores individuales, que participan en las sesiones de juego activas en la ubicación de la flota.

Sesiones de juego

La pestaña Game sessions (Sesiones de juego) enumera las sesiones de juego pasadas y presentes alojadas en la flota, incluida información detallada. Elija un ID de sesión de juego para acceder a la información adicional de la sesión de juego, incluidas las sesiones de jugador. Para obtener más información sobre las sesiones de jugador, consulte Visualización de datos de sesiones de juego y de jugador.

Visualización de datos de sesiones de juego y de jugador

Puede visualizar información sobre las sesiones de juego y sobre cada jugador. Para obtener más información acerca de las sesiones de juego y de jugador, consulte Cómo se conectan los jugadores a los juegos.

Para visualizar los datos de sesiones de juego y de jugador, realice el siguiente procedimiento:

1. En la consola de Amazon GameLift, en el panel de navegación, elija Flotas.
2. Elija la flota de la lista de Flotas que aloja las sesiones de juego.

3. Seleccione la pestaña Sesiones de juego. Esta pestaña enumera todas las sesiones de juego alojadas en la flota junto con información resumida.
4. Elija una sesión de juego para ver información adicional sobre la sesión de juego y una lista de los jugadores conectados al juego.

Detalles

Información general

En esta sección se muestra un resumen de la información de la sesión de juego.

- Estado: estado de la sesión de juego.
  - Activando: la instancia inicia una sesión de juego.
  - Activo: se está ejecutando una sesión de juego y está disponible para recibir jugadores, en función de la política de creación de jugadores de la sesión.
  - Terminado: ha finalizado la sesión de juego.
- ARN: nombre de recurso de Amazon (ARN) del grupo de usuario.
- Nombre: nombre generado para la sesión de juego.
- Ubicación: ubicación en la que Amazon GameLift alojó la sesión de juego.
- Hora de creación: fecha y hora en que Amazon GameLift creó la sesión de transmisión.
- Hora de finalización: fecha y hora en que finalizó la sesión de juego.
- Nombre de DNS: nombre del host de la sesión de juego.
- Dirección IP: dirección IP específica de la sesión de juego.
- Puerto: número de puerto que se utiliza para conectar con la sesión de juego.
- ID de creador: identificador único del jugador que inició la sesión de juego.
- Política de creación de sesiones del jugador: indica si la sesión de juego acepta nuevos jugadores.
- Política de protección para el escalado de juegos: tipo de protección de sesión de juego que se establecerá en todas las instancias nuevas que Amazon GameLift inicie en la flota.

Datos de juegos

Datos con un formato correcto para enviarlos a la sesión de juego al inicio.

Propiedades del juego

Propiedades de pares clave y de valores que influyen en la sesión de juego.

Datos de emparejamiento

El JSON del emparejador FlexMatch. Para revisar y editar el emparejador, elija Ver configuración del emparejador Para obtener más información sobre el emparejamiento de FlexMatch, consulte Creación de un emparejador.

Sesiones de jugador

Para cada sesión de juego se recopilan los siguientes datos de sesiones de jugador:

- ID de sesión del jugador: el identificador asignado a la sesión del jugador.
- ID de jugador: identificador exclusivo para el jugador. Elija este ID para obtener información adicional del jugador.
- Estado: estado de la sesión de jugador. Los posibles estados son los siguientes:
  - Reservada: la sesión de jugador está reservada, pero el jugador no se ha conectado todavía.
  - Activa: la sesión de jugador está conectada en el servidor de juegos.
  - Completada: la sesión de jugador ha finalizado y el jugador ya no está conectado.
  - Tiempo de espera agotado: el jugador no pudo conectarse.
- Hora de creación: hora en que el jugador se conectó a la sesión de juego.
- Hora de finalización: hora en que el jugador se desconectó de la sesión de juego.
- Datos del jugador: información sobre el jugador proporcionada durante la creación de la sesión del jugador.

Información sobre el jugador

Puede ver información adicional de un jugador seleccionado, incluida una lista de todos los juegos a los que se ha conectado el jugador en todas las flotas de la región actual. Esta información incluye el estado, la hora de inicio y de fin, y el tiempo total de conexión de cada sesión de jugador. Elija ver los datos de las sesiones de juego y de las flotas relevantes.

Visualización de alias

La página Alias muestra información sobre los alias de flota creados en su región actual. Para ver la página Alias, elija Alias en el panel de navegación.

En la página Alias, puede realizar las siguientes acciones:

- Crear un alias nuevo. Elija Crear alias.
- Filtrar y ordenar la tabla de alias. Utilice los controles en la parte superior de la tabla.
- Ver los detalles de los alias. Elija un nombre del alias para abrir la página de detalles del alias.
- Eliminar un alias. Elija un alias y, a continuación, elija Eliminar.

Detalles de alias

La página Detalles de alias muestra información resumida sobre el alias.

En esta página puede hacer lo siguiente:

- Editar un alias. Elija Edit (Editar).
- Ver las flotas asociadas al alias.
- Eliminar un alias. Elija Eliminar (Delete).

La información detallada del alias incluye:

- ID: número exclusivo que se utiliza para identificar al alias.
- Descripción: descripción del alias.
- ARN: nombre de recurso de Amazon (ARN) de un alias.
- Creación: es la fecha y la hora en que se creó el alias.
- Última actualización: fecha y la hora en la que se actualizó por última vez el alias.
- Tipo de direccionamiento: opción de direccionamiento para el alias, que puede ser una de las siguientes:
  - Simple: redirige el tráfico de jugadores a un ID de la flota especificado. Puede actualizar el ID de flota de un alias en cualquier momento.
  - Terminal: transmite un mensaje al cliente. Por ejemplo, puede redirigir a los jugadores que estén usando un cliente obsoleto a una ubicación en la que puedan obtener una actualización.
- Etiquetas: pares de clave y valor que se utilizan para identificar el alias.

Visualización de colas

Puede consultar información sobre todas las colas de ubicación de sesiones de juego existentes. La página Colas muestra las colas creadas en la región actual. En la página Queues, puede crear una cola nueva, eliminar las colas existentes o abrir una página de detalles de una cola seleccionada. Una página de detalles de la cola contiene la configuración y los datos de métricas de la cola. Para obtener más información sobre las colas, consulte Configuración de colas de Amazon GameLift para la ubicación de las sesiones de juego.

La página Colas muestra la siguiente información resumida de cada cola:

- Nombre de la cola: el nombre asignado a la cola. Las solicitudes de sesiones de juego nuevas utilizan este nombre para identificar una cola.
- Tiempo de espera de la cola: duración máxima, en segundos, que una solicitud de ubicación de sesión de juego se mantiene en la cola antes de expirar.
- Destinos en la cola: número de flotas que aparece en la lista de flotas de la configuración de la cola. Amazon GameLift coloca nuevas sesiones de juego en cualquier flota en la cola.

Visualización de los detalles de la cola

Puede acceder a información detallada sobre cualquier cola, incluida la configuración y las métricas de la cola. Para abrir una página de detalles de la cola, vaya a la página Colas y elija el nombre de la misma.

La página de detalles de la cola muestra una tabla de resumen y pestañas que contienen información adicional. En esta página puede hacer lo siguiente:

- Actualizar la configuración de la cola, la lista de destinos y las políticas de latencia de los jugadores. Elija Edit (Editar).
- Eliminar una cola. Después de eliminar una cola, todas las solicitudes de sesiones de juego nuevas que hacen referencia al nombre de la cola darán error. Elija Eliminar (Delete).

> **Note**
>
> Para restaurar una cola eliminada, cree una nueva con el nombre de la cola eliminada.

Detalles

Información general

La sección Información general muestra el Nombre de recurso de Amazon (ARN) y el tiempo de espera de la cola. Puede usar el ARN al hacer referencia a la cola en otras acciones o áreas de Amazon GameLift. El tiempo de espera es la duración máxima, en segundos, que una solicitud de ubicación de sesión de juego se mantiene en la cola antes de expirar.

Notificación de eventos

La sección Notificación de eventos muestra el tema de SNS en el que Amazon GameLift publica las notificaciones de eventos y los datos de eventos que se añaden a todos los eventos creados por esa cola.

Etiquetas

La tabla de Etiquetas muestra las claves y los valores utilizados para etiquetar el recurso. Para obtener más información sobre el etiquetado, consulte Etiquetado de recursos de AWS.

Métricas

La pestaña Metrics muestra una representación gráfica de las métricas de la cola a lo largo del tiempo.

Las métricas de la cola incluyen un amplio abanico de información que describe la actividad de ubicación de toda la cola, incluidas las ubicaciones correctas organizadas por región. Puede utilizar los datos de la región para saber dónde alojar los juegos. Las métricas de ubicación regional pueden ayudar a detectar problemas con el diseño general de las colas.

Las métricas de las cola también están disponibles en Amazon CloudWatch. Para ver descripciones de métricas disponibles, consulte Métricas de Amazon GameLift para colas.

Destinos

La pestaña Destinations muestra todas las flotas o alias correspondientes a la cola.

Cuando Amazon GameLift busca los destinos para los recursos disponibles para alojar una nueva sesión de juego, la búsqueda la realiza según el orden predeterminado indicado aquí. Mientras exista capacidad en el primer destino de la lista, Amazon GameLift colocará ahí las sesiones de juego nuevas. Puede hacer que determinadas solicitudes de ubicación de sesiones de juego anulen el

orden predeterminado si proporciona los datos de latencia de los jugadores. Esos datos indican a Amazon GameLift que busque un destino disponible que tenga la latencia media de jugadores más baja. Para obtener más información sobre el diseño de colas, consulte Diseño de colas de sesiones de juego.

Ubicación de la sesión

Políticas de latencia de jugadores

La sección Políticas de latencia de jugadores muestra todas las políticas que utiliza la cola. En las tablas aparecen las políticas en el orden en que se aplican.

Locations

La sección Ubicaciones muestra las ubicaciones en las que esta cola puede incluir una sesión de juego.

Priority (Prioridad)

La sección Prioridad muestra el orden en que la cola evalúa los detalles de una sesión de juego.

Orden de ubicación

La sección Orden de ubicación muestra el orden predeterminado que utiliza la cola al colocar las sesiones de juego. La cola usa ese orden si no ha definido otros tipos de prioridad.

Supervisión de Amazon GameLift

Si utiliza Amazon GameLift FleetIQ como función independiente con Amazon EC2, consulte Seguridad en Amazon EC2 en la Guía del usuario de Amazon EC2.

La supervisión es una parte importante del mantenimiento de la fiabilidad, la disponibilidad y el rendimiento de Amazon GameLift y sus demás AWS soluciones. Las métricas de Amazon tienen tres usos principales GameLift: monitorear el estado del sistema y configurar alarmas, rastrear el rendimiento y el uso de los servidores de juegos y administrar la capacidad mediante el escalado manual o el autoscaling.

AWS proporciona las siguientes herramientas de supervisión para vigilar Amazon GameLift, informar cuando algo va mal y tomar medidas automáticas cuando sea necesario:

- GameLift Consola Amazon
- Amazon CloudWatch: puedes monitorear GameLift las métricas de Amazon en tiempo real, así como las métricas de otros AWS recursos y aplicaciones que ejecutas en AWS los servicios. CloudWatch ofrece un conjunto de funciones de monitoreo, que incluyen herramientas para crear paneles personalizados y la posibilidad de configurar alarmas que notifican o toman medidas cuando una métrica alcanza un umbral específico.
- AWS CloudTrail— captura todas las llamadas a la API y los eventos relacionados realizados por o en nombre de tu AWS cuenta para Amazon GameLift y otros AWS servicios. Los datos se entregan como archivos de registro en el bucket de Amazon S3 que especifique. Puedes identificar qué usuarios y cuentas llamaron AWS, la dirección IP de origen desde la que se realizaron las llamadas y cuándo se produjeron.
- Registros de sesiones de juego: puede enviar mensajes de servidor personalizados para sus sesiones de juego a archivos de registro almacenados en Amazon S3.

Temas

Supervisión de Amazon GameLift con Amazon CloudWatch

Puede supervisar Amazon GameLift mediante Amazon CloudWatch, un servicio de AWS que recopila y procesa datos sin procesar y los procesa en métricas legibles y prácticamente en tiempo real. Estas estadísticas se conservan durante 15 meses para proporcionar una perspectiva histórica sobre el rendimiento del alojamiento del servidor de juegos con Amazon GameLift. Puede establecer alarmas que vigilen determinados umbrales y enviar notificaciones o realizar acciones cuando se alcancen dichos umbrales. Para obtener más información, consulte la Guía del usuario de Amazon CloudWatch.

En las siguientes tablas se muestran las métricas y dimensiones de Amazon GameLift. Todas las métricas que están disponibles en CloudWatch también están disponibles en la consola de Amazon GameLift, que proporciona los datos como un conjunto de gráficos personalizables. Para tener acceso a las métricas de CloudWatch para los juegos, utilice la AWS Management Console, la AWS CLI o la API de CloudWatch.

Si una métrica no tiene una ubicación, utiliza la ubicación de inicio.

Dimensiones de las métricas de Amazon GameLift

Amazon GameLift permite filtrar métricas por las siguientes dimensiones.

Dimensión	Descripción
`Location`	Permite filtrar las métricas para una ubicación de implementación de la flota. Si una métrica no tiene una ubicación, utiliza la ubicación de inicio.
`FleetId`	Filtrar métricas para una sola flota. Esta dimensión se puede usar con todas las métricas de instancias, procesos de servidor, sesiones de juego y sesiones de jugador.
`MetricGroup`	Filtrar métricas para una colección de flotas. Agregue una flota a un grupo de métricas de flota agregando el nombre del grupo de métricas a los atributos de la flota (consulte UpdateFleetAttributes()). Esta dimensión se puede usar con todas las métricas de

Dimensión	Descripción
	instancias, procesos de servidor, sesiones de juego y sesiones de jugador.
`QueueName`	Filtrar métricas para una sola cola. Esta dimensión se usa únicamente con métricas para las colas de sesiones de juego.
`ConfigurationName`	Filtrar métricas para una única configuración de emparejamiento. Esta dimensión se usa con métricas de configuraciones de emparejamiento.
`ConfigurationName-RuleName`	Filtrar métricas para una intersección de una configuración de emparejamiento y una regla de emparejamiento. Esta dimensión solo se usa con métricas de reglas de emparejamiento.
`InstanceType`	Filtrar métricas para una designación de tipo de instancia EC2, como "c4.large". Esta dimensión se usa con métricas de instancias de spot.
`OperatingSystem`	Filtrar métricas para el sistema operativo de una instancia Esta dimensión se utiliza con métricas de instancias de spot.
`GameServerGroup`	Permite filtrar métricas de FleetIQ para un grupo de servidores de juegos.

Métricas de Amazon GameLift para flotas

El espacio de nombres `AWS/GameLift` incluye las siguientes métricas relativas a la actividad de una flota o grupo de flotas. Las flotas se utilizan con una solución de Amazon GameLift administrada. El servicio de Amazon GameLift envía métricas a CloudWatch cada minuto.

Instancias

Métrica	Descripción
`ActiveInstances`	Instancias con el estado ACTIVE, lo que indica que se trata de procesos del servidor activos. El recuento incluye las instancias inactivas y las que alojan una o varias sesiones de juego. Esta métrica mide la capacidad total actual de las instancias. Se puede usar con el escalado automático. Unidades: recuento Estadísticas de CloudWatch pertinentes: Average, Minimum, Maximum Dimensiones: ubicación
`DesiredInstances`	Número de destino de instancias activas que utiliza Amazon GameLift para mantener la flota. Con el escalado automático, este valor se determina en función de las políticas de escalado aplicadas actualmente. Sin el escalado automático, este valor se establece manualmente. Esta métrica no está disponible cuando se consultan datos de grupos de métricas de flotas. Unidades: recuento Estadísticas de CloudWatch pertinentes: Average, Minimum, Maximum Dimensiones: ubicación
`IdleInstances`	Instancias activas que alojan actualmente cero (0) sesiones de juego. Esta métrica mide la capacidad disponible pero sin usar. Se puede usar con el escalado automático.

Métrica	Descripción
	Unidades: recuento Estadísticas de CloudWatch pertinentes: Average, Minimum, Maximum Dimensiones: ubicación
`MaxInstances`	Número máximo de instancias permitidas para la flota. El número máximo de instancias de una flota determina el límite máximo de capacidad durante el escalado ascendente manual o automático. Esta métrica no está disponible cuando se consultan datos de grupos de métricas de flotas. Unidades: recuento Estadísticas de CloudWatch pertinentes: Average, Minimum, Maximum Dimensiones: ubicación
`MinInstances`	Número mínimo de instancias permitidas para la flota. El número mínimo de instancias de una flota determina el límite inferior de capacidad durante el escalado descendente manual o automático. Esta métrica no está disponible cuando se consultan datos de grupos de métricas de flotas. Unidades: recuento Estadísticas de CloudWatch pertinentes: Average, Minimum, Maximum Dimensiones: ubicación

Métrica	Descripción
`PercentIdleInstances`	Porcentaje de todas las instancias activas que se encuentran inactivas (calculado como `IdleInstances / ActiveInstances`). Esta métrica se puede usar para el escalado automático. Unidades: porcentaje Estadísticas de CloudWatch pertinentes: Average, Minimum, Maximum Dimensiones: ubicación
`RecycledInstances`	Número de instancias de spot que se han reciclado y reemplazado. Amazon GameLift recicla las instancias de spot que actualmente no alojan sesiones de juego y que tienen una alta probabilidad de interrupción. Unidades: recuento Estadísticas de CloudWatch pertinentes: Sum, Average, Minimum, Maximum Dimensiones: ubicación
`InstanceInterruptions`	Número de las instancias de spot que se han interrumpido. Unidades: recuento Estadísticas de CloudWatch pertinentes: Sum, Average, Minimum, Maximum Dimensiones: ubicación

Métrica	Descripción
CPUUtilization	Métricas de EC2. En el caso de Amazon GameLift, esta métrica representa el rendimiento del hardware en todas las instancias activas de una ubicación de flota. El porcentaje de tiempo de CPU física que Amazon EC2 utiliza para ejecutar la instancia , que incluye el tiempo dedicado a ejecutar tanto el código de usuario como el código de Amazon EC2. Las herramientas de su sistema operativo pueden mostrar un porcentaje diferente al de CloudWatch debido a factores como la simulación de dispositi vos heredados, la configuración de dispositivos no heredados, las cargas de trabajo con muchas interrupciones, la migración en vivo y la actualización en vivo. Unidades: porcentaje
NetworkIn	Métricas de EC2. En el caso de Amazon GameLift, esta métrica representa el rendimiento del hardware en todas las instancias activas de una ubicación de flota. El número de bytes recibidos en todas las interfaces de red por la instancia. Esta métrica identifica el volumen de tráfico de red entrante para una aplicación en una sola instancia. Unidades: bytes
NetworkOut	Métricas de EC2. En el caso de Amazon GameLift, esta métrica representa el rendimiento del hardware en todas las instancias activas de una ubicación de flota. El número de bytes enviados en todas las interfaces de red por la instancia. Esta métrica identifica el volumen de tráfico de red saliente para una aplicación en una sola instancia. Unidades: bytes

Métrica	Descripción
DiskReadBytes	Métricas de EC2. En el caso de Amazon GameLift, esta métrica representa el rendimiento del hardware en todas las instancias activas de una ubicación de flota. Bytes leídos de todos los volúmenes del almacén de instancias disponibles para la instancia . Esta métrica se usa para determinar el volumen de datos que la aplicación lee del disco duro de la instancia. Se puede usar para determinar la velocidad de la aplicación. Unidades: bytes
DiskWriteBytes	Métricas de EC2. En el caso de Amazon GameLift, esta métrica representa el rendimiento del hardware en todas las instancias activas de una ubicación de flota. Bytes escritos en todos los volúmenes del almacén de instancias disponibles para la instancia . Esta métrica se usa para determinar el volumen de datos que la aplicación escribe en el disco duro de la instancia. Se puede usar para determinar la velocidad de la aplicación. Unidades: bytes
DiskReadOps	Métricas de EC2. En el caso de Amazon GameLift, esta métrica representa el rendimiento del hardware en todas las instancias activas de una ubicación de flota. Operaciones de lectura completadas de todos los volúmenes del almacén de instancias disponibles para la instancia en un periodo de tiempo especific ado. Para calcular el promedio de operaciones de E/S por segundo (IOPS) para el periodo, divida el total de operaciones del periodo por el número de segundos de ese periodo. Unidades: recuento

Métrica	Descripción
DiskWriteOps	Métricas de EC2. En el caso de Amazon GameLift, esta métrica representa el rendimiento del hardware en todas las instancias activas de una ubicación de flota. Operaciones de escritura completadas en todos los volúmenes del almacén de instancias disponibles para la instancia en un periodo de tiempo especific ado. Para calcular el promedio de operaciones de E/S por segundo (IOPS) para el periodo, divida el total de operaciones del periodo por el número de segundos de ese periodo. Unidades: recuento

Procesos del servidor

Métrica	Descripción
ActiveServerProcesses	Procesos del servidor con el estado ACTIVE, lo que indica que se están ejecutando y pueden alojar sesiones de juego. El recuento incluye los procesos de servidor inactivos y los que alojan sesiones de juego. Esta métrica mide la capacidad total actual de los procesos del servidor. Unidades: recuento Estadísticas de CloudWatch pertinentes: Average, Minimum, Maximum Dimensiones: ubicación
HealthyServerProcesses	Procesos del servidor activos registrados como en buen estado. Esta métrica es útil para realizar un seguimiento del estado general de los servidores de juego de la flota.

Métrica	Descripción
	Unidades: recuento Estadísticas de CloudWatch pertinentes: Average, Minimum, Maximum Dimensiones: ubicación
`PercentHealthyServerProcesses`	Porcentaje de todos los procesos del servidor activos registrados como en buen estado (calculado como `HealthyServerProcesses / ActiveServerProcesses`). Unidades: porcentaje Estadísticas de CloudWatch pertinentes: Average, Minimum, Maximum Dimensiones: ubicación
`ServerProcessAbnormalTerminations`	Procesos del servidor que se cerraron debido a circunstancias anormales desde el último informe. Esta métrica incluye las terminaciones iniciadas por el servicio de Amazon GameLift. Esto se produce cuando un proceso del servidor deja de responder, registra una y otra vez comprobaciones de estado no superadas o no se termina limpiamente (al llamar a ProcessEnding()). Unidades: recuento Estadísticas de CloudWatch pertinentes: Sum, Average, Minimum, Maximum Dimensiones: ubicación

Métrica	Descripción
ServerProcessActivations	Procesos del servidor que pasaron correctamente del estado ACTIVATING a ACTIVE desde el último informe. Los procesos del servidor no pueden alojar sesiones de juego hasta que están activos. Unidades: recuento Estadísticas de CloudWatch pertinentes: Sum, Average, Minimum, Maximum Dimensiones: ubicación
ServerProcessTerminations	Procesos del servidor que se cerraron desde el último informe. Esto incluye todos los procesos del servidor que por algún motivo pasaron a tener el estado TERMINATED, incluidas las terminaciones del proceso normales y anormales. Unidades: recuento Estadísticas de CloudWatch pertinentes: Sum, Average, Minimum, Maximum Dimensiones: ubicación

Sesiones de juego

Métrica	Descripción
ActivatingGameSessions	Sesiones de juego con el estado ACTIVATING, lo que indica que se están iniciando. Las sesiones de juego no pueden alojar jugadores hasta que están activas. Un número elevado durante un periodo de tiempo prolongado puede indicar que las sesiones de juego no están pasando del estado ACTIVATIN

Métrica	Descripción
	G al estado ACTIVE. Se puede usar con el escalado automático. Unidades: recuento Estadísticas de CloudWatch pertinentes: Average, Minimum, Maximum Dimensiones: ubicación
`ActiveGameSessions`	Sesiones de juego con el estado ACTIVE, lo que indica que pueden alojar jugadores y que están alojando cero o más jugadores. Esta métrica mide el número total de sesiones de juego alojadas actualmente. Se puede usar con el escalado automático. Unidades: recuento Estadísticas de CloudWatch pertinentes: Average, Minimum, Maximum Dimensiones: ubicación

Métrica	Descripción
`AvailableGameSessions`	Procesos de servidor activos y en buen estado que no se utilizan actualmente para alojar una sesión de juego y que pueden iniciar una nueva sesión de juego sin retrasos para activar nuevos procesos o instancias del servidor. Se puede usar con el escalado automático. **Note** Para las flotas que limitan las activaciones simultáneas de sesiones de juego, utilice la métrica `ConcurrentActivatableGameSessions`. Esa métrica representa con mayor precisión el número de sesiones de juego nuevas que se pueden iniciar sin ningún tipo de retraso. Unidades: recuento Estadísticas de CloudWatch pertinentes: Average, Minimum, Maximum Dimensiones: ubicación

Métrica	Descripción
`ConcurrentActivatableGameSessions`	Procesos de servidor activos y en buen estado que no se utilizan actualmente para alojar una sesión de juego y que pueden iniciar inmediatamente una nueva sesión de juego. Esta métrica se diferencia de `AvailableGameSessions` de la siguiente manera: no incluye los procesos del servidor que actualmente no pueden activar una nueva sesión de juego debido a los límites en las activaciones de las sesiones de juego. Consulte la configuración opcional de RuntimeConfiguration `MaxConcurrentGameSessionActivations` de la flota. En el caso de las flotas que no limitan las activaciones de las sesiones de juego, la métrica es idéntica a `AvailableGameSessions` . Unidades: recuento Estadísticas de CloudWatch pertinentes: Average, Minimum, Maximum Dimensiones: ubicación

Métrica	Descripción
`PercentAvailableGameSessions`	Porcentaje de slots de sesiones de juego en todos los procesos del servidor activos (en buen estado o en mal estado) que no se están usando actualmente (calculado como `AvailableGameSessions / [ActiveGameSessions + AvailableGameSessions + unhealthy server processes]`). Se puede usar con el escalado automático. Unidades: porcentaje Estadísticas de CloudWatch relevantes: Average Dimensiones: ubicación
`GameSessionInterruptions`	Número de sesiones de juego en instancias de spot que se han interrumpido. Unidades: recuento Estadísticas de CloudWatch pertinentes: Sum, Average, Minimum, Maximum Dimensiones: ubicación

Sesiones de jugador

Métrica	Descripción
`CurrentPlayerSessions`	Sesiones de jugador con el estado ACTIVE (el jugador está conectado a una sesión de juego activa) o con el estado RESERVED (al jugador se le ha asignado un slot en una sesión de juego pero aún no se ha conectado). Se puede usar con el escalado automático.

Métrica	Descripción
	Unidades: recuento Estadísticas de CloudWatch pertinentes: Average, Minimum, Maximum
`PlayerSessionActivations`	Sesiones de jugador que pasaron correctamente del estado RESERVED a ACTIVE desde el último informe. Esto ocurre cuando un jugador se conecta correctamente a una sesión de juego activa. Unidades: recuento Estadísticas de CloudWatch pertinentes: Sum, Average, Minimum, Maximum

Métricas de Amazon GameLift para colas

El espacio de nombres `Amazon GameLift` incluye las siguientes métricas relativas a la actividad en una cola de ubicación de sesión de juego. Las colas se utilizan con una solución de Amazon GameLift administrada. El servicio de Amazon GameLift envía métricas a CloudWatch cada minuto.

Métrica	Descripción
`AverageWaitTime`	Promedio de tiempo que las solicitudes de ubicación de sesión de juego en la cola con el estado PENDING han esperado a ser atendidas. Unidades: segundos Estadísticas pertinentes de CloudWatch: Average, Minimum, Maximum, Sum Dimensiones: ubicación
`FirstChoiceNotViable`	Sesiones de juego que se transfirieron correctam ente pero NO a la primera flota elegida, ya que dicha flota no se consideraba viable (por ejemplo, una

Métrica	Descripción
	flota de spot con un gran número de interrupciones). Esta métrica se basa en el costo, no en la latencia. La primera flota elegida es la primera flota que aparece en la cola o, si una solicitud de ubicación contiene datos sobre la latencia de los jugadores, es la primera flota elegida por la priorización de FleetIQ. Si no hay flotas de spot viables, se puede seleccion ar cualquier flota en esa región. Unidades: recuento Estadísticas pertinentes de CloudWatch: Average, Minimum, Maximum, Sum
`FirstChoiceOutOfCapacity`	Sesiones de juego que se transfieren correctam ente pero NO a la primera flota elegida, ya que dicha flota no tiene recursos disponibles. La primera flota elegida es la primera flota que aparece en la cola o, si una solicitud de ubicación contiene datos sobre la latencia de los jugadores, es la primera flota elegida por la priorización de FleetIQ definida. Unidades: recuento Estadísticas pertinentes de CloudWatch: Average, Minimum, Maximum, Sum

Métrica	Descripción
`LowestLatencyPlacement`	Sesiones de juego que se transfirieron correctamente a una región que ofrece la menor latencia posible en la cola para los jugadores. Esta métrica solamente se emite solo cuando los datos de latencia de los jugadores se incluyen en la solicitud de ubicación. Unidades: recuento Estadísticas pertinentes de CloudWatch: Average, Minimum, Maximum, Sum
`LowestPricePlacement`	Sesiones de juego que se transfirieron correctamente a una flota que tiene el menor precio posible de la cola para la región elegida. Esta flota puede ser una flota de spot o, si la cola no tiene instancias de spot, una instancia bajo demanda. Esta métrica solamente se emite solo cuando los datos de latencia de los jugadores se incluyen en la solicitud de ubicación. Unidades: recuento Estadísticas pertinentes de CloudWatch: Average, Minimum, Maximum, Sum
`Placement <region name>`	Sesiones de juego que se transfirieron correctamente a las flotas situadas en la región especificada. Esta métrica desglosa el valor `PlacementsSucceeded` por regiones. Unidades: recuento Estadísticas de CloudWatch relevantes: Sum

Métrica	Descripción
PlacementsCanceled	Solicitudes de ubicación de sesión de juego que se cancelaron porque se agotó el tiempo de espera desde el último informe. Unidades: recuento Estadísticas pertinentes de CloudWatch: Average, Minimum, Maximum, Sum
PlacementsFailed	Solicitudes de ubicación de sesión de juego que no se realizaron correctamente por alguna razón desde el último informe. Unidades: recuento Estadísticas pertinentes de CloudWatch: Average, Minimum, Maximum, Sum
PlacementsStarted	Solicitudes de ubicación de sesión de juego nuevas que se añadieron a la cola desde el último informe. Unidades: recuento Estadísticas pertinentes de CloudWatch: Average, Minimum, Maximum, Sum
PlacementsSucceeded	Solicitudes de ubicación de sesión de juego que dieron lugar a una nueva sesión de juego desde el último informe. Unidades: recuento Estadísticas pertinentes de CloudWatch: Average, Minimum, Maximum, Sum

Métrica	Descripción
`PlacementsTimedOut`	Solicitudes de ubicación de sesión de juego que alcanzaron el límite de tiempo de espera de la cola sin ser atendidas desde el último informe. Unidades: recuento Estadísticas pertinentes de CloudWatch: Average, Minimum, Maximum, Sum
`QueueDepth`	Número de solicitudes de ubicación de sesión de juego de la cola con el estado PENDING. Unidades: recuento Estadísticas pertinentes de CloudWatch: Average, Minimum, Maximum, Sum Dimensiones: ubicación

Métricas de Amazon GameLift para el emparejamiento

El espacio de nombres de `Amazon GameLift` contiene las métricas de la actividad de emparejamiento sobre las configuraciones y las reglas de esta actividad. El emparejamiento de FlexMatch se utiliza con una solución de Amazon GameLift administrada. El servicio de Amazon GameLift envía métricas a CloudWatch cada minuto.

Para obtener más información sobre la secuencia de la actividad de emparejamiento, consulte Funcionamiento de Amazon GameLift FlexMatch.

Configuraciones de emparejamiento

Métrica	Descripción
`CurrentTickets`	En la actualidad, se están procesando solicitud es de emparejamiento o están a la espera de ser procesadas.

Métrica	Descripción
	Unidades: recuento Estadísticas pertinentes de CloudWatch: Average, Minimum, Maximum, Sum
`MatchAcceptancesTimedOut`	Para las configuraciones de emparejamiento que requieren aceptación, los posibles emparejamientos que agotaron su tiempo durante la aceptación desde el último informe. Unidades: recuento Estadísticas de CloudWatch relevantes: Sum
`MatchesAccepted`	Para las configuraciones de emparejamiento que requieren aceptación, los posibles emparejamientos que se aceptaron desde el último informe. Unidades: recuento Estadísticas de CloudWatch relevantes: Sum
`MatchesCreated`	Emparejamientos potenciales creados desde el último informe. Unidades: recuento Estadísticas de CloudWatch relevantes: Sum
`MatchesPlaced`	Emparejamientos que se ubicaron en una sesión de juego desde el último informe. Unidades: recuento Estadísticas de CloudWatch relevantes: Sum

Métrica	Descripción
MatchesRejected	Para las configuraciones de emparejamiento que requieren aceptación, los posibles emparejamientos que rechazó al menos un jugador desde el último informe. Unidades: recuento Estadísticas de CloudWatch relevantes: Sum
PlayersStarted	Jugadores en incidencias de emparejamiento que se añadieron desde el último informe. Unidades: recuento Estadísticas de CloudWatch relevantes: Sum
TicketsFailed	Solicitudes de emparejamiento con error desde el último informe. Unidades: recuento Estadísticas de CloudWatch relevantes: Sum
TicketsStarted	Nuevas solicitudes de emparejamiento creadas desde el último informe. Unidades: recuento Estadísticas de CloudWatch relevantes: Sum
TicketsTimedOut	Solicitudes de emparejamiento que alcanzaron el límite de tiempo de espera desde el último informe. Unidades: recuento Estadísticas de CloudWatch relevantes: Sum

Métrica	Descripción
TimeToMatch	Para las solicitudes de emparejamiento que se pusieron en un emparejamiento potencial antes del último informe, la cantidad de tiempo entre la creación de la incidencia y la creación de emparejamientos potenciales. Unidades: segundos Estadísticas de CloudWatch relevantes: Muestras de datos, Media, Mínimo y Máximo
TimeToTicketCancel	Para las solicitudes de emparejamiento que se cancelaron antes del último informe, la cantidad de tiempo entre la creación de la incidencia y la cancelación. Unidades: segundos Estadísticas de CloudWatch relevantes: Muestras de datos, Media, Mínimo y Máximo
TimeToTicketSuccess	Para las solicitudes de emparejamiento con éxito antes del último informe, la cantidad de tiempo entre la creación de la incidencia y la ubicación de emparejamiento con éxito. Unidades: segundos Estadísticas de CloudWatch relevantes: Muestras de datos, Media, Mínimo y Máximo

Reglas de emparejamiento

Métrica	Descripción
`RuleEvaluationsPassed`	Evaluaciones de reglas durante el proceso de emparejamiento superadas desde el último informe. Esta métrica se limita a las 50 primeras reglas. Unidades: recuento Estadísticas de CloudWatch relevantes: Sum
`RuleEvaluationsFailed`	Evaluaciones de reglas durante el emparejamiento que no se superaron desde el último informe. Esta métrica se limita a las 50 primeras reglas. Unidades: recuento Estadísticas de CloudWatch relevantes: Sum

Métricas de Amazon GameLift para FleetIQ

El espacio de nombres de `Amazon GameLift` incluye métricas para el grupo de servidores de juegos y la actividad del servidor de juegos de FleetIQ como parte de una solución de FleetIQ independiente para el alojamiento de juegos. El servicio de Amazon GameLift envía métricas a CloudWatch cada minuto. Consulte también Supervisión de las instancias y grupos de escalado automático mediante Amazon CloudWatch en la Guía del usuario de escalado automático de Amazon EC2.

Métrica	Descripción
`AvailableGameServers`	Servidores para videojuegos que están disponibles para ejecutar una ejecución de juego y en los que actualmente no hay actividad de juego. Este número incluye servidores para videojuegos que se han reclamado pero todavía se encuentran en estado AVAILABLE.

Métrica	Descripción
	Unidades: recuento Estadísticas de CloudWatch relevantes: Sum Dimensiones: GameServerGroup
`UtilizedGameServers`	Servidores para videojuegos en los que actualmente hay actividad de juego. Este número incluye los servidores de juegos que están en estado UTILIZADOS. Unidades: recuento Estadísticas de CloudWatch relevantes: Sum Dimensiones: GameServerGroup
`DrainingAvailableGameServers`	Servidores para videojuegos en instancias programadas para la terminación que actualmente no son compatibles con la actividad de juego. Estos servidores para videojuegos tienen la prioridad más baja de reclamación en respuesta a una nueva solicitud de reclamación. Unidades: recuento Estadísticas de CloudWatch relevantes: Sum Dimensiones: GameServerGroup
`DrainingUtilizedGameServers`	Servidores para videojuegos en instancias programadas para la terminación que actualmente admiten la actividad de juego. Unidades: recuento Estadísticas de CloudWatch relevantes: Sum Dimensiones: GameServerGroup

Métrica	Descripción
`PercentUtilizedGameServers`	Parte de los servidores para videojuegos que actualmente admiten ejecuciones de juegos. Esta métrica indica la cantidad de capacidad del servidor para videojuegos que se está utilizando actualmente. Es útil para habilitar una política de Auto Scaling que agregue y elimine dinámicamente instancias para satisfacer la demanda de jugadores. Unidades: porcentaje Estadísticas de CloudWatch pertinentes: Average, Minimum, Maximum Dimensiones: GameServerGroup
`GameServerInterruptions`	Servidores para videojuegos en instancias de spot que se interrumpieron debido a una disponibilidad de spot limitada. Unidades: recuento Estadísticas de CloudWatch relevantes: Sum Dimensiones: GameServerGroup, InstanceType
`InstanceInterruptions`	Instancias de spot que se interrumpieron debido a una disponibilidad limitada. Unidades: recuento Estadísticas de CloudWatch relevantes: Sum Dimensiones: GameServerGroup, InstanceType

Registro de llamadas a la API de Amazon GameLift con AWS CloudTrail

Amazon GameLift se integra con AWS CloudTrail, un servicio que proporciona un registro de las acciones que realiza un usuario, un rol o un servicio de AWS en Amazon GameLift. CloudTrail captura todas las llamadas a la API para Amazon GameLift como eventos. Las llamadas capturadas incluyen las llamadas desde la consola de Amazon GameLift y las llamadas desde el código a las operaciones de la API de Amazon GameLift. Si crea un registro de seguimiento, puede habilitar la entrega continua de eventos de CloudTrail a un bucket de Amazon S3, incluidos los eventos de Amazon GameLift. Si no configura un registro de seguimiento, puede ver los eventos más recientes de la consola de CloudTrail en el Historial de eventos. Mediante la información recopilada por CloudTrail, puede determinar la solicitud que se realizó a Amazon GameLift, la dirección IP de origen desde la que se realizó la solicitud, quién realizó la solicitud, cuándo se realizó y otros detalles adicionales.

Para obtener más información acerca de CloudTrail, consulte la Guía del usuario de AWS CloudTrail.

Información de Amazon GameLift en CloudTrail

CloudTrail se habilita en su Cuenta de AWS cuando la crea. Cuando se produce una actividad en Amazon GameLift, dicha actividad se registra en un evento de CloudTrail junto con los eventos de los demás servicios de AWS en Historial de eventos. Puede ver, buscar y descargar los últimos eventos de la Cuenta de AWS. Para obtener más información, consulte Ver eventos con el historial de eventos de CloudTrail.

Para mantener un registro continuo de los eventos de la Cuenta de AWS, incluidos los eventos de Amazon GameLift, cree un registro de seguimiento. Un registro de seguimiento permite a CloudTrail enviar archivos de registro a un bucket de Amazon S3. De forma predeterminada, cuando se crea un registro de seguimiento en la consola, el registro de seguimiento se aplica a todas las Regiones de AWS. El registro de seguimiento registra los eventos de todas las regiones de la partición de AWS y envía los archivos de registro al bucket de Amazon S3 especificado. También es posible configurar otros servicios de AWS para analizar en profundidad y actuar en función de los datos de eventos recopilados en los registros de CloudTrail. Para obtener más información, consulte los siguientes temas:

- Introducción a la creación de registros de seguimiento
- Servicios e integraciones compatibles con CloudTrail

- Configuración de notificaciones de Amazon SNS para CloudTrail
- Recibir archivos de registro de CloudTrail de varias regiones y Recibir archivos de registro de CloudTrail de varias cuentas

Todas las acciones de Amazon GameLift se registran en CloudTrail y están documentadas en la Referencia de la API de Amazon GameLift. Por ejemplo, las llamadas a las acciones `CreateGameSession`, `CreatePlayerSession` y `UpdateGameSession` generan entradas en los archivos de registros de CloudTrail.

Cada entrada de registro o evento contiene información sobre quién generó la solicitud. La información de identidad del usuario le ayuda a determinar lo siguiente:

- Si la solicitud se realizó con credenciales de usuario AWS Identity and Access Management (IAM) o credenciales de usuario raíz.
- Si la solicitud se realizó con credenciales de seguridad temporales de un rol o fue un usuario federado.
- Si la solicitud la realizó otro servicio de AWS.

Para obtener más información, consulte el elemento userIdentity de CloudTrail.

Descripción de las entradas de archivos de registro de Amazon GameLift

Un registro de seguimiento es una configuración que permite la entrega de eventos como archivos de registros en un bucket de Amazon S3 que especifique. Los archivos log de CloudTrail pueden contener una o varias entradas de log. Un evento representa una solicitud específica realizada desde un origen y contiene información sobre la acción solicitada, la fecha y la hora de la acción, los parámetros de la solicitud, etc. Los archivos de registro de CloudTrail no rastrean el orden en la pila de las llamadas públicas a la API, por lo que estas no aparecen en ningún orden específico.

En el ejemplo siguiente, se muestra una entrada de registro de CloudTrail que ilustra las acciones `CreateFleet` y `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"
        },
    ]
}
```

Registro de mensajes del servidor en Amazon GameLift

Puede capturar mensajes de servidor personalizados de los servidores de Amazon GameLift en archivos de registro. La forma en que configure el registro depende de si utiliza servidores personalizados o Realtime Servers (consulte las subsecciones correspondientes en este capítulo).

Temas

Registro de mensajes del servidor (servidores personalizados)

Puedes capturar mensajes de servidor personalizados de tus servidores GameLift personalizados de Amazon en archivos de registro. Para obtener más información sobre el registro en Realtime Servers, consulte Registro de mensajes del servidor (Realtime Servers).

> **Important**
>
> Hay un límite en el tamaño de un archivo de registro por sesión de juego (consulta los GameLift puntos finales y las cuotas de Amazon en Referencia general de AWS). Cuando finaliza una sesión de juego, Amazon GameLift carga los registros del servidor en Amazon Simple Storage Service (Amazon S3). Amazon GameLift no subirá registros que superen el límite. Los registros pueden crecer muy rápido y superar el límite de tamaño. Debe supervisar los registros y limitar la salida del registro a los mensajes necesarios únicamente.

Configuración del registro para servidores personalizados

Con los servidores GameLift personalizados de Amazon, escribes tu propio código para realizar el registro, que configuras como parte de la configuración del proceso del servidor. Amazon GameLift utiliza su configuración de registro para identificar los archivos que debe cargar en Amazon S3 al final de cada sesión de juego.

Las siguientes instrucciones muestran cómo configurar el registro mediante ejemplos de código simplificados:

C++

Para configurar el registro (C++), realice el siguiente procedimiento:

1. Cree un vector de cadenas que sean rutas de directorio a los archivos de registro del servidor de juegos.

```
std::string serverLog("serverOut.log");        // Example server log file
std::vector<std::string> logPaths;
logPaths.push_back(serverLog);
```

2. Proporcione su vector como el LogParametersde su ProcessParametersobjeto.

```
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. Proporcione el ProcessParametersobjeto cuando llame a ProcessReady().

```
Aws::GameLift::GenericOutcome outcome =
   Aws::GameLift::Server::ProcessReady(processReadyParameter);
```

Para ver un ejemplo más completo, consulte ProcessReady().

C#

Para configurar el registro (C#), realice el siguiente procedimiento:

1. Cree una lista de cadenas que sean rutas de directorio a los archivos de registro del servidor de juegos.

```
List<string> logPaths = new List<string>();
logPaths.Add("C:\\game\\serverOut.txt");     // Example of a log file that the
 game server writes
```

2. Proporcione su lista como la LogParametersde su ProcessParametersobjeto.

```
var processReadyParameter = new ProcessParameters(
    this.OnGameSession,
    this.OnProcessTerminate,
    this.OnHealthCheck,
    this.OnGameSessionUpdate,
    port,
    new LogParameters(logPaths));
```

3. Proporcione el ProcessParametersobjeto cuando llame a ProcessReady().

```
var processReadyOutcome =
   GameLiftServerAPI.ProcessReady(processReadyParameter);
```

Para ver un ejemplo más completo, consulte ProcessReady().

Escritura en los registros

Los archivos de registro existirán después de que comience el proceso del servidor. Puede escribir en los registros mediante cualquier método para escribir en los archivos. Para capturar todos los resultados estándar y de errores del servidor, reasigne los flujos de salida a los archivos de registro, como en los ejemplos siguientes:

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"));
```

Acceso a los registros del servidor

Cuando finaliza una sesión de juego, Amazon almacena GameLift automáticamente los registros en un bucket de Amazon S3 y los conserva durante 14 días. Para obtener la ubicación de los registros

de una sesión de juego, puedes usar la operación de la GetGameSessionLogUrlAPI. Para descargar los registros, utilice la URL que devuelve la operación.

Registro de mensajes del servidor (Realtime Servers)

Puede capturar mensajes de servidor personalizados de sus Realtime Servers en archivos de registro. Para obtener más información sobre el registro en servidores personalizados, consulte Registro de mensajes del servidor (servidores personalizados).

Existen distintos tipos de mensajes que puede enviar a sus archivos de registro (consulte Registro de los mensajes en el script del servidor). Además de los mensajes personalizados, los Realtime Servers generan mensajes del sistema con los mismos tipos de mensajes y los escriben en los mismos archivos de registro. Puede ajustar el nivel de registro de su flota para reducir la cantidad de mensajes de registro que generan sus servidores (consulte Ajuste del nivel de registro).

> **Important**
>
> Hay un límite en el tamaño de un archivo de registro por sesión de juego (consulta los GameLift puntos finales y las cuotas de Amazon en Referencia general de AWS). Cuando finaliza una sesión de juego, Amazon GameLift carga los registros del servidor en Amazon Simple Storage Service (Amazon S3). Amazon GameLift no subirá registros que superen el límite. Los registros pueden crecer muy rápido y superar el límite de tamaño. Debe supervisar los registros y limitar el resultado del registro a los mensajes necesarios únicamente.

Registro de los mensajes en el script del servidor

Puede generar mensajes personalizados en el script de sus Realtime Servers. Siga estos pasos para enviar mensajes del servidor a un archivo de registro:

1. Cree una variable para retener la referencia en el objeto del registrador.

   ```
   var logger;
   ```

2. En la función `init()`, obtenga el registrador del objeto de sesión y asígnelo a la variable del registrador.

   ```
   function init(rtSession) {
   ```

```
    session = rtSession;
    logger = session.getLogger();
}
```

3. Llame a la función apropiada del registrador para generar un mensaje.

   Mensajes de depuración

   ```
   logger.debug("This is my debug message...");
   ```

   Mensajes informativos

   ```
   logger.info("This is my info message...");
   ```

   Mensajes de advertencia

   ```
   logger.warn("This is my warn message...");
   ```

   Mensajes de error

   ```
   logger.error("This is my error message...");
   ```

   Mensajes de error grave

   ```
   logger.fatal("This is my fatal error message...");
   ```

   Mensajes de error grave en la experiencia del cliente

   ```
   logger.cxfatal("This is my customer experience fatal error message...");
   ```

Para ver un ejemplo de las instrucciones de registro en un script, consulte Ejemplo del script de Realtime Servers.

El resultado de los archivos de registro indica el tipo de mensaje (`DEBUG`, `INFO`, `WARN`, `ERROR`, `FATAL` y `CXFATAL`), como se muestra en las siguientes líneas de un registro de ejemplo:

```
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.
```

Acceso a los registros del servidor

Cuando finaliza una sesión de juego, Amazon almacena GameLift automáticamente los registros en Amazon S3 y los conserva durante 14 días. Puedes usar la llamada a la GetGameSessionLogUrl API para obtener la ubicación de los registros de una sesión de juego. Utilice la URL devuelta por la llamada a la API para descargar los registros.

Ajuste del nivel de registro

Los registros pueden crecer muy rápido y superar el límite de tamaño. Debe supervisar los registros y limitar el resultado del registro a los mensajes necesarios únicamente. En el caso de Realtime Servers, puede ajustar el nivel de registro mediante un parámetro en la configuración del tiempo de ejecución de la flota en forma de `loggingLevel:`*`LOGGING_LEVEL`*, donde *`LOGGING_LEVEL`* sea uno de los siguientes valores:

1. `debug`
2. `info` (predeterminado)
3. `warn`
4. `error`
5. `fatal`
6. `cxfatal`

Esta lista está ordenada de menos grave (`debug`) a más grave (`cxfatal`). Si establece un parámetro `loggingLevel` único, el servidor solo registrará los mensajes con ese nivel de gravedad o uno superior. Por ejemplo, si se establece `loggingLevel:error`, todo los servidores de su flota solo escribirán mensajes `error`, `fatal` y `cxfatal` en el registro.

Puede establecer el nivel de registro de flota al crearlo o una vez que esté en ejecución. Si cambia el nivel de registro de la flota después de que esté en ejecución, solo se verán afectados los registros de las sesiones de juego creadas después de la actualización. Los registros de las sesiones de juego existentes no se verán afectados. Si no establece un nivel de registro al crear la flota,

los servidores establecerán el nivel de registro en `info` de forma predeterminada. Consulte las siguientes secciones para obtener instrucciones sobre cómo configurar el nivel de registro.

Configuración del nivel de registro al crear una flota de Realtime Servers (consola)

Siga las instrucciones que aparecen en Creación de una flota administrada por Amazon GameLift para crear su flota y añada lo siguiente:

- En el subpaso Asignación de los procesos del servidor del paso de Administración de procesos, proporcione el par clave-valor del nivel de registro (por ejemplo, `loggingLevel:error`) como valor para Parámetros de lanzamiento. Utilice un carácter no alfanumérico (excepto coma) para separar el nivel de registro de cualquier parámetro adicional (por ejemplo, `loggingLevel:error +map Winter444`).

Configuración del nivel de registro al crear una flota de Realtime Servers (AWS CLI)

Siga las instrucciones que aparecen en Creación de una flota administrada por Amazon GameLift para crear su flota y añada lo siguiente:

- En el argumento del parámetro `--runtime-configuration` para `create-fleet`, proporcione el par clave-valor del nivel de registro (por ejemplo, `loggingLevel:error`) como valor para `Parameters`. Utilice un carácter no alfanumérico (excepto coma) para separar el nivel de registro de cualquier parámetro adicional. Vea el siguiente ejemplo:

```
--runtime-configuration "GameSessionActivationTimeoutSeconds=60,
                    MaxConcurrentGameSessionActivations=2,
                    ServerProcesses=[{LaunchPath=/local/game/myRealtimeLaunchScript.js,
                                      Parameters=loggingLevel:error +map Winter444,
                                      ConcurrentExecutions=10}]"
```

Configuración del nivel de registro para una flota de Realtime Servers en ejecución (consola)

Sigue las instrucciones que aparecen en Actualización de la configuración de una flota para actualizar tu flota mediante Amazon GameLift Console y añade lo siguiente:

- En la página Editar flota, en Asignación de los procesos del servidor, proporcione el par clave-valor del nivel de registro (por ejemplo`loggingLevel:error`) como valor para los parámetros de lanzamiento. Utilice un carácter no alfanumérico (excepto coma) para separar el nivel de registro de cualquier parámetro adicional (por ejemplo, `loggingLevel:error +map Winter444`).

Configuración del nivel de registro para una flota de Realtime Servers en ejecución (AWS CLI)

Siga las instrucciones que aparecen en Actualización de la configuración de una flota para actualizar su flota mediante la AWS CLI y añada lo siguiente:

- En el argumento del parámetro `--runtime-configuration` para `update-runtime-configuration`, proporcione el par clave-valor del nivel de registro (por ejemplo, `loggingLevel:error`) como valor para `Parameters`. Utilice un carácter no alfanumérico (excepto coma) para separar el nivel de registro de cualquier parámetro adicional. Vea el siguiente ejemplo:

```
--runtime-configuration "GameSessionActivationTimeoutSeconds=60,
                    MaxConcurrentGameSessionActivations=2,
                    ServerProcesses=[{LaunchPath=/local/game/myRealtimeLaunchScript.js,
                                      Parameters=loggingLevel:error +map Winter444,
                                      ConcurrentExecutions=10}]"
```

Seguridad en Amazon GameLift

Si utiliza Amazon GameLift FleetIQ como función independiente con Amazon EC2, consulte Seguridad en Amazon EC2 en la Guía del usuario de Amazon EC2.

La seguridad en la nube es la máxima prioridad. AWS Como cliente de AWS , se beneficiará de una arquitectura de red y de centros de datos diseñados para satisfacer los requisitos de seguridad de las organizaciones más exigentes.

La seguridad es una responsabilidad compartida entre usted AWS y usted. El modelo de responsabilidad compartida la describe como seguridad de la nube y seguridad en la nube:

- Seguridad de la nube: AWS es responsable de proteger la infraestructura que ejecuta AWS los servicios en la AWS nube. AWS también le proporciona servicios que puede utilizar de forma segura. Los auditores externos prueban y verifican periódicamente la eficacia de nuestra seguridad como parte de los AWS programas de de . Para obtener más información sobre los programas de conformidad que se aplican a Amazon GameLift, consulta AWS los servicios incluidos en el ámbito de aplicación por programa de conformidad y .
- Seguridad en la nube: tu responsabilidad viene determinada por el AWS servicio que utilices. También eres responsable de otros factores, como la confidencialidad de tus datos, los requisitos de tu empresa y las leyes AWS y reglamentos aplicables.

Esta documentación te ayuda a entender cómo aplicar el modelo de responsabilidad compartida cuando utilizas Amazon GameLift. En los temas siguientes, se muestra cómo configurar Amazon GameLift para que cumpla con sus objetivos de seguridad y conformidad. También aprenderás a utilizar otros AWS servicios que te ayudan a supervisar y proteger tus GameLift recursos de Amazon.

Temas

- Mejores prácticas de seguridad para Amazon GameLift

Protección de datos en Amazon GameLift

Si utiliza Amazon GameLift FleetIQ como función independiente con Amazon EC2, consulte Seguridad en Amazon EC2 en la Guía del usuario de Amazon EC2.

El AWS modelo de se aplica a protección de datos en Amazon GameLift. Como se describe en este modelo, AWS es responsable de proteger la infraestructura global en la que se ejecutan todos los Nube de AWS. Usted es responsable de mantener el control sobre el contenido alojado en esta infraestructura. Usted también es responsable de las tareas de administración y configuración de seguridad para los Servicios de AWS que utiliza. Para obtener más información sobre la privacidad de los datos, consulte las Preguntas frecuentes sobre la privacidad de datos. Para obtener información sobre la protección de datos en Europa, consulte la publicación de blog sobre el Modelo de responsabilidad compartida de AWS y GDPR en el Blog de seguridad de AWS .

Con fines de protección de datos, le recomendamos que proteja Cuenta de AWS las credenciales y configure los usuarios individuales con AWS IAM Identity Center o AWS Identity and Access Management (IAM). De esta manera, solo se otorgan a cada usuario los permisos necesarios para cumplir sus obligaciones laborales. También recomendamos proteger sus datos de la siguiente manera:

- Utilice la autenticación multifactor (MFA) en cada cuenta.
- Utilice SSL/TLS para comunicarse con los recursos. AWS Se recomienda el uso de TLS 1.2 y recomendamos TLS 1.3.
- Configure la API y el registro de actividad de los usuarios con. AWS CloudTrail
- Utilice soluciones de AWS cifrado, junto con todos los controles de seguridad predeterminados Servicios de AWS.
- Utilice servicios de seguridad administrados avanzados, como Amazon Macie, que lo ayuden a detectar y proteger los datos confidenciales almacenados en Amazon S3.
- Si necesita módulos criptográficos validados por FIPS 140-2 para acceder a AWS través de una interfaz de línea de comandos o una API, utilice un punto final FIPS. Para obtener más información sobre los puntos de conexión de FIPS disponibles, consulte Estándar de procesamiento de la información federal (FIPS) 140-2.

Se recomienda encarecidamente no introducir nunca información confidencial o sensible, como, por ejemplo, direcciones de correo electrónico de clientes, en etiquetas o campos de formato libre, tales como el campo Nombre. Esto incluye cuando trabajas con Amazon GameLift u otros Servicios de AWS usuarios mediante la consola, la API o AWS los SDK. AWS CLI Cualquier dato que ingrese en etiquetas o campos de formato libre utilizados para nombres se puede emplear para los registros de facturación o diagnóstico. Si proporciona una URL a un servidor externo, recomendamos encarecidamente que no incluya información de credenciales en la URL a fin de validar la solicitud para ese servidor.

Los datos GameLift específicos de Amazon se gestionan de la siguiente manera:

- Las compilaciones de servidores de juegos y los scripts que subas a Amazon se GameLift almacenan en Amazon S3. El cliente no tiene acceso directo a estos datos una vez cargados. Un usuario autorizado puede obtener acceso temporal para cargar archivos, pero no puede verlos ni actualizarlos en Amazon S3 directamente. Para eliminar scripts y compilaciones, usa la GameLift consola de Amazon o la API de servicio.
- Los datos de registro de la sesión de juego se almacenan en Amazon S3 durante un periodo limitado una vez completada la sesión de juego. Los usuarios autorizados pueden acceder a los datos de registro descargándolos a través de un enlace en la GameLift consola de Amazon o mediante llamadas a la API del servicio.
- Los datos de métricas y eventos se almacenan en Amazon GameLift y se puede acceder a ellos a través de la GameLift consola de Amazon o mediante llamadas a la API del servicio. Los datos pueden recuperarse en flotas, instancias, ubicaciones de sesiones de juego, tickets de emparejamiento, sesiones de juego y sesiones de jugador. También se puede acceder a los datos a través de Amazon CloudWatch y CloudWatch Events.
- Los datos proporcionados por los clientes se almacenan en Amazon. GameLift Los usuarios autorizados pueden obtener acceso a ellos mediante llamadas a la API de servicio. La información potencialmente confidencial puede incluir datos del jugador, datos de sesiones de jugador y de juego (incluida la información de conexión) y datos del creador de emparejamiento, entre otros.

> **Note**
>
> Si proporciona identificadores de jugadores personalizados en las solicitudes, se espera que sean valores UUID anónimos y que no contengan información identificativa del jugador.

Para obtener más información sobre la protección de datos, consulte la entrada de blog relativa al modelo de responsabilidad compartida de AWS y GDPR en el blog de seguridad de AWS .

Cifrado en reposo

El cifrado en reposo de los datos GameLift específicos de Amazon se gestiona de la siguiente manera:

- Las compilaciones y los scripts del servidor de juegos se almacenan en buckets de Amazon S3 con cifrado del servidor.
- Los datos proporcionados por los clientes se almacenan GameLift en Amazon en un formato cifrado.

Cifrado en tránsito

Las conexiones a las GameLift API de Amazon se realizan a través de una conexión segura (SSL) y se autentican mediante la versión 4 de AWS Signature (cuando se conecta a través de la AWS CLI o el AWS SDK, la firma se gestiona automáticamente). La autenticación se administra mediante las políticas de acceso definidas por IAM para las credenciales de seguridad que se utilizan a fin de realizar la conexión.

La comunicación directa entre los clientes y servidores de juego es la siguiente:

- En el caso de los servidores de juegos personalizados alojados en GameLift los recursos de Amazon, la comunicación no implica el GameLift servicio de Amazon. El cifrado de esta comunicación es responsabilidad del cliente. Puede utilizar flotas habilitadas con TLS para que sus clientes de juegos autentiquen el servidor de juegos al conectarse y para cifrar toda comunicación entre el cliente y el servidor de juegos.
- Para Servidores en tiempo real con la generación de certificados TLS habilitada, el tráfico entre el cliente de juegos y los servidores en tiempo real que utilizan el SDK del cliente en tiempo real se cifra en tránsito. El tráfico TCP se cifra con TLS 1.2 y el tráfico UDP se cifra con DTLS 1.2.

Privacidad del tráfico entre redes

Puede acceder de forma remota a sus GameLift instancias de Amazon de forma segura. Para las instancias que utilizan Linux, SSH proporciona un canal de comunicaciones seguro para el acceso remoto. Para las instancias que ejecutan Windows, utilice un cliente de protocolo de escritorio

remoto (RDP). Con Amazon GameLift FleetiQ, el acceso remoto a sus instancias mediante Systems AWS Manager Session Manager y Run Command se cifra mediante TLS 1.2, y las solicitudes para crear una conexión se firman mediante SiGv4. Si necesitas ayuda para conectarte a una GameLift instancia de Amazon gestionada, consultaConéctese remotamente a las instancias de GameLift la flota de Amazon.

Gestión de identidades y accesos para Amazon GameLift

AWS Identity and Access Management (IAM) es un Servicio de AWS que ayuda a los administradores a controlar de forma segura el acceso a los recursos de AWS. Los administradores de IAM controlan quién puede autenticarse (iniciar sesión) y quién puede autorizarse (tener permisos) para usar los recursos de Amazon GameLift . IAM es un servicio de Servicio de AWS que se puede utilizar sin cargo adicional.

Temas

Público

La forma de usar AWS Identity and Access Management (IAM) varía según el trabajo que realices en Amazon GameLift.

Usuario del servicio: si utilizas el GameLift servicio de Amazon para realizar tu trabajo, el administrador te proporcionará las credenciales y los permisos que necesitas. A medida que utilices más GameLift funciones de Amazon para realizar tu trabajo, es posible que necesites permisos adicionales. Entender cómo se administra el acceso puede ayudarlo a solicitar los permisos correctos al administrador. Si no puedes acceder a una función de Amazon GameLift, consultaSolución de problemas de GameLift identidad y acceso a Amazon.

Administrador de servicios: si estás a cargo de GameLift los recursos de Amazon en tu empresa, probablemente tengas acceso total a Amazon GameLift. Es tu trabajo determinar a qué GameLift funciones y recursos de Amazon deben acceder los usuarios de tu servicio. Luego, debe enviar

solicitudes a su administrador de IAM para cambiar los permisos de los usuarios de su servicio. Revise la información de esta página para conocer los conceptos básicos de IAM. Para obtener más información sobre cómo su empresa puede utilizar IAM con Amazon GameLift, consulteCómo GameLift funciona Amazon con IAM.

Administrador de IAM: si eres administrador de IAM, quizá te interese obtener más información sobre cómo puedes redactar políticas para gestionar el acceso a Amazon. GameLift Para ver ejemplos de políticas GameLift basadas en la identidad de Amazon que puedes usar en IAM, consulta. Ejemplos de políticas basadas en identidad para Amazon GameLift

Autenticación con identidades

La autenticación es la manera de iniciar sesión en AWS mediante credenciales de identidad. Debe estar autenticado (haber iniciado sesión en AWS) como Usuario raíz de la cuenta de AWS, como un usuario de IAM o asumiendo un rol de IAM.

Puede iniciar sesión en AWS como una identidad federada mediante las credenciales proporcionadas a través de una fuente de identidad de AWS IAM Identity Center. Los usuarios (del IAM Identity Center), la autenticación de inicio de sesión único de su empresa y sus credenciales de Google o Facebook son ejemplos de identidades federadas. Al iniciar sesión como una identidad federada, su administrador habrá configurado previamente la federación de identidades mediante roles de IAM. Cuando accede a AWS mediante la federación, está asumiendo un rol de forma indirecta.

Según el tipo de usuario que sea, puede iniciar sesión en la AWS Management Console o en el portal de acceso a AWS. Para obtener más información sobre el inicio de sesión en AWS, consulte Cómo iniciar sesión en su Cuenta de AWS en la Guía del usuario de AWS Sign-In.

Si accede a AWS mediante programación, AWS proporciona un kit de desarrollo de software (SDK) y una interfaz de la línea de comandos (CLI) para firmar criptográficamente las solicitudes mediante el uso de las credenciales. Si no utiliza las herramientas de AWS, debe firmar usted mismo las solicitudes. Para obtener más información sobre la firma de solicitudes, consulte Firma de solicitudes API de AWS en la Guía del usuario de IAM.

Independientemente del método de autenticación que utilice, es posible que deba proporcionar información de seguridad adicional. Por ejemplo, AWS le recomienda el uso de la autenticación multifactor (MFA) para aumentar la seguridad de su cuenta. Para más información, consulte Autenticación multifactor en la Guía del usuario de AWS IAM Identity Center y Uso de la autenticación multifactor (MFA) en AWS en la Guía del usuario de IAM.

Usuario raíz de Cuenta de AWS

Cuando se crea una Cuenta de AWS, se comienza con una identidad de inicio de sesión que tiene acceso completo a todos los recursos y Servicios de AWS de la cuenta. Esta identidad recibe el nombre de usuario raíz de la Cuenta de AWS y se accede a ella iniciando sesión con el email y la contraseña que utilizó para crear la cuenta. Recomendamos encarecidamente que no utilice el usuario raíz para sus tareas diarias. Proteja las credenciales del usuario raíz y utilícelas solo para las tareas que solo el usuario raíz pueda realizar. Para ver la lista completa de las tareas que requieren que inicie sesión como usuario raíz, consulte Tareas que requieren credenciales de usuario raíz en la Guía del usuario de IAM.

Identidad federada

Como práctica recomendada, solicite que los usuarios humanos, incluidos los que requieren acceso de administrador, utilicen la federación con un proveedor de identidades para acceder a Servicios de AWS utilizando credenciales temporales.

Una identidad federada es un usuario del directorio de usuarios de su empresa, un proveedor de identidad web, el AWS Directory Service, el directorio del Identity Center, o cualquier usuario que acceda a Servicios de AWS utilizando credenciales proporcionadas a través de una fuente de identidad. Cuando identidades federadas acceden a las Cuentas de AWS, asumen roles y los roles proporcionan credenciales temporales.

Para una administración de acceso centralizada, le recomendamos que utilice AWS IAM Identity Center. Puede crear usuarios y grupos en el IAM Identity Center o puede conectarse y sincronizar con un conjunto de usuarios y grupos de su propia fuente de identidad para usarlos en todas sus aplicaciones y Cuentas de AWS. Para más información, consulte ¿Qué es IAM Identity Center? en la Guía del usuario de AWS IAM Identity Center.

Usuarios y grupos de IAM

Un usuario de IAM es una identidad en su Cuenta de AWS que dispone de permisos específicos para una sola persona o aplicación. Siempre que sea posible, recomendamos emplear credenciales temporales, en lugar de crear usuarios de IAM que tengan credenciales de larga duración como contraseñas y claves de acceso. No obstante, si tiene casos de uso específicos que requieran credenciales de larga duración con usuarios de IAM, recomendamos rotar las claves de acceso. Para más información, consulte Rotar las claves de acceso periódicamente para casos de uso que requieran credenciales de larga duración en la Guía del usuario de IAM.

Un grupo de IAM es una identidad que especifica un conjunto de usuarios de IAM. No puede iniciar sesión como grupo. Puede usar los grupos para especificar permisos para varios usuarios a la vez. Los grupos facilitan la administración de los permisos de grandes conjuntos de usuarios. Por ejemplo, podría tener un grupo cuyo nombre fuese IAMAdmins y conceder permisos a dicho grupo para administrar los recursos de IAM.

Los usuarios son diferentes de los roles. Un usuario se asocia exclusivamente a una persona o aplicación, pero la intención es que cualquier usuario pueda asumir un rol que necesite. Los usuarios tienen credenciales permanentes a largo plazo y los roles proporcionan credenciales temporales. Para más información, consulte Cuándo crear un usuario de IAM (en lugar de un rol) en la Guía del usuario de IAM.

Roles de IAM

Un rol de IAM es una identidad de tu Cuenta de AWS que dispone de permisos específicos. Es similar a un usuario de IAM, pero no está asociado a una determinada persona. Puede asumir temporalmente un rol de IAM en la AWS Management Console cambiando de roles. Puede asumir un rol llamando a una operación de AWS CLI o de la API de AWS, o utilizando una URL personalizada. Para más información sobre los métodos para el uso de roles, consulte Uso de roles de IAM en la Guía del usuario de IAM.

Los roles de IAM con credenciales temporales son útiles en las siguientes situaciones:

- Acceso de usuario federado: para asignar permisos a una identidad federada, puede crear un rol y definir sus permisos. Cuando se autentica una identidad federada, se asocia la identidad al rol y se le conceden los permisos define el rol. Para obtener información acerca de roles para federación, consulte Creación de un rol para un proveedor de identidades de terceros en la Guía del usuario de IAM. Si utiliza el IAM Identity Center, debe configurar un conjunto de permisos. El Centro de identidades de IAM correlaciona el conjunto de permisos con un rol en IAM para controlar a qué pueden acceder sus identidades después de autenticarse. Para obtener información acerca de los conjuntos de permisos, consulte Conjuntos de permisos en la Guía del usuario de AWS IAM Identity Center.
- Permisos de usuario de IAM temporales: un usuario de IAM puede asumir un rol de IAM para recibir temporalmente permisos distintos que le permitan realizar una tarea concreta.
- Acceso entre cuentas: puede utilizar un rol de IAM para permitir que alguien (una entidad principal de confianza) de otra cuenta acceda a los recursos de la cuenta. Los roles son la forma principal de conceder acceso entre cuentas. No obstante, con algunos Servicios de AWS se puede asociar una política directamente a un recurso (en lugar de utilizar un rol como representante). Para

obtener información acerca de la diferencia entre los roles y las políticas basadas en recursos para el acceso entre cuentas, consulte Cómo los roles de IAM difieren de las políticas basadas en recursos en la Guía del usuario de IAM.

- Acceso entre servicios: algunos Servicios de AWS utilizan características de otros Servicios de AWS. Por ejemplo, cuando realiza una llamada en un servicio, es común que ese servicio ejecute aplicaciones en Amazon EC2 o almacene objetos en Amazon S3. Es posible que un servicio haga esto usando los permisos de la entidad principal, usando un rol de servicio o usando un rol vinculado a servicios.
  - Reenviar sesiones de acceso (FAS): cuando utiliza un rol o un usuario de IAM para llevar a cabo acciones en AWS, se le considera una entidad principal. Cuando utiliza algunos servicios, es posible que realice una acción que desencadene otra acción en un servicio diferente. FAS utiliza los permisos de la entidad principal para llamar a un Servicio de AWS, combinados con el Servicio de AWS solicitante para realizar solicitudes a servicios posteriores. Las solicitudes de FAS solo se realizan cuando un servicio recibe una solicitud que requiere interacciones con otros Servicios de AWS o recursos para completarse. En este caso, debe tener permisos para realizar ambas acciones. Para obtener información sobre las políticas a la hora de realizar solicitudes de FAS, consulte Reenviar sesiones de acceso.
  - Rol de servicio: un rol de servicio es un rol de IAM que adopta un servicio para realizar acciones en su nombre. Un administrador de IAM puede crear, modificar y eliminar un rol de servicio desde IAM. Para más información, consulte Creación de un rol para delegar permisos a un Servicio de AWS en la Guía del usuario de IAM.
  - Rol vinculado al servicio: un rol vinculado al servicio es un tipo de rol de servicio que está vinculado a un Servicio de AWS. El servicio puede asumir el rol para realizar una acción en su nombre. Los roles vinculados a servicios aparecen en la Cuenta de AWS y son propiedad del servicio. Un administrador de IAM puede ver, pero no editar, los permisos de los roles vinculados a servicios.
- Aplicaciones que se ejecutan en Amazon EC2: puede utilizar un rol de IAM que le permita administrar credenciales temporales para las aplicaciones que se ejecutan en una instancia de EC2 y realizan solicitudes a la AWS CLI o a la API de AWS. Es preferible hacerlo de este modo a almacenar claves de acceso en la instancia EC2. Para asignar un rol de AWS a una instancia de EC2 y ponerla a disposición de todas las aplicaciones, cree un perfil de instancia asociado a la instancia. Un perfil de instancia contiene el rol y permite a los programas que se ejecutan en la instancia EC2 obtener credenciales temporales. Para más información, consulte Uso de un rol de IAM para conceder permisos a aplicaciones que se ejecutan en instancias Amazon EC2 en la Guía del usuario de IAM.

Para obtener información sobre el uso de los roles de IAM, consulte Cuándo crear un rol de IAM (en lugar de un usuario) en la Guía del usuario de IAM.

Administración de acceso mediante políticas

Para controlar el acceso en AWS, se crean políticas y se adjuntan a identidades o recursos de AWS. Una política es un objeto de AWS que, cuando se asocia a una identidad o un recurso, define sus permisos. AWS evalúa estas políticas cuando una entidad principal (sesión de rol, usuario o usuario raíz) realiza una solicitud. Los permisos en las políticas determinan si la solicitud se permite o se deniega. La mayoría de las políticas se almacenan en AWS como documentos JSON. Para obtener más información sobre la estructura y el contenido de los documentos de política JSON, consulte Información general de políticas JSON en la Guía del usuario de IAM.

Los administradores pueden utilizar las políticas JSON de AWS para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

De forma predeterminada, los usuarios y los roles no tienen permisos. Para conceder permiso a los usuarios para realizar acciones en los recursos que necesiten, un administrador puede crear políticas de IAM. A continuación, el administrador puede añadir las políticas de IAM a roles y los usuarios pueden asumirlos.

Las políticas de IAM definen permisos para una acción independientemente del método que se utilice para realizar la operación. Por ejemplo, suponga que dispone de una política que permite la acción `iam:GetRole`. Un usuario con dicha política puede obtener información del rol de la AWS Management Console, la AWS CLI o la API de AWS.

Políticas basadas en identidades

Las políticas basadas en identidad son documentos de políticas de permisos JSON que puede asociar a una identidad, como un usuario de IAM, un grupo de usuarios o un rol. Estas políticas controlan qué acciones pueden realizar los usuarios y los roles, en qué recursos y en qué condiciones. Para obtener más información sobre cómo crear una política basada en identidad, consulte Creación de políticas de IAM en la Guía del usuario de IAM.

Las políticas basadas en identidades pueden clasificarse además como políticas insertadas o políticas administradas. Las políticas insertadas se integran directamente en un único usuario, grupo o rol. Las políticas administradas son políticas independientes que puede asociar a varios usuarios, grupos y roles de su Cuenta de AWS. Las políticas administradas incluyen las políticas administradas de AWS y las políticas administradas por el cliente. Para más información sobre cómo elegir una

política administrada o una política insertada, consulte Elegir entre políticas administradas y políticas insertadas en la Guía del usuario de IAM.

Políticas basadas en recursos

Las políticas basadas en recursos son documentos de política JSON que se asocian a un recurso. Ejemplos de políticas basadas en recursos son las políticas de confianza de roles de IAM y las políticas de bucket de Amazon S3. En los servicios que admiten políticas basadas en recursos, los administradores de servicios pueden utilizarlos para controlar el acceso a un recurso específico. Para el recurso al que se asocia la política, la política define qué acciones puede realizar una entidad principal especificada en ese recurso y en qué condiciones. Debe especificar una entidad principal en una política en función de recursos. Las entidades principales pueden incluir cuentas, usuarios, roles, usuarios federados o Servicios de AWS.

Las políticas basadas en recursos son políticas insertadas que se encuentran en ese servicio. No se puede utilizar políticas de IAM administradas de AWS en una política basada en recursos.

Listas de control de acceso (ACL)

Las listas de control de acceso (ACL) controlan qué entidades principales (miembros de cuentas, usuarios o roles) tienen permisos para acceder a un recurso. Las ACL son similares a las políticas basadas en recursos, aunque no utilizan el formato de documento de políticas JSON.

Amazon S3, AWS WAF y Amazon VPC son ejemplos de servicios que admiten las ACL. Para obtener más información sobre las ACL, consulte Información general de Lista de control de acceso (ACL) en la Guía para desarrolladores de Amazon Simple Storage Service.

Otros tipos de políticas

AWS admite otros tipos de políticas adicionales menos frecuentes. Estos tipos de políticas pueden establecer el máximo de permisos que los tipos de políticas más frecuentes le conceden.

- Límites de permisos: un límite de permisos es una característica avanzada que le permite establecer los permisos máximos que una política basada en identidad puede conceder a una entidad de IAM (usuario o rol de IAM). Puede establecer un límite de permisos para una entidad. Los permisos resultantes son la intersección de las políticas basadas en la identidad de la entidad y los límites de permisos. Las políticas basadas en recursos que especifique el usuario o rol en el campo `Principal` no estarán restringidas por el límite de permisos. Una denegación explícita en cualquiera de estas políticas anulará el permiso. Para obtener más información sobre los límites

de los permisos, consulte Límites de permisos para las entidades de IAM en la Guía del usuario de IAM.

- Políticas de control de servicio (SCP): las SCP son políticas de JSON que especifican los permisos máximos de una empresa o una unidad organizativa en AWS Organizations. AWS Organizations es un servicio que le permite agrupar y administrar de manera centralizada varias Cuentas de AWS que posea su empresa. Si habilita todas las características en una empresa, entonces podrá aplicar políticas de control de servicio (SCP) a una o todas sus cuentas. Una SCP limita los permisos para las entidades de las cuentas de miembros, incluido cada Usuario raíz de la cuenta de AWS. Para más información sobre Organizations y las SCP, consulte Funcionamiento de las SCP en la Guía del usuario de AWS Organizations.
- Políticas de sesión: las políticas de sesión son políticas avanzadas que se pasan como parámetro cuando se crea una sesión temporal mediante programación para un rol o un usuario federado. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades del rol y las políticas de la sesión. Los permisos también pueden proceder de una política en función de recursos. Una denegación explícita en cualquiera de estas políticas anulará el permiso. Para más información, consulte Políticas de sesión en la Guía del usuario de IAM.

Varios tipos de políticas

Cuando se aplican varios tipos de políticas a una solicitud, los permisos resultantes son más complicados de entender. Para obtener información acerca de cómo AWS decide si permitir o no una solicitud cuando hay varios tipos de políticas implicados, consulte Lógica de evaluación de políticas en la Guía del usuario de IAM.

Cómo GameLift funciona Amazon con IAM

Antes de utilizar IAM para gestionar el acceso a Amazon GameLift, consulta qué funciones de IAM están disponibles para su uso con Amazon. GameLift

Funciones de IAM que puedes usar con Amazon GameLift

Característica de IAM	GameLift Soporte de Amazon
Políticas basadas en identidades	Sí
Políticas basadas en recursos	No
Acciones de políticas	Sí

Característica de IAM	GameLift Soporte de Amazon
Recursos de políticas	Sí
Claves de condición de política (específicas del servicio)	Sí
ACL	No
ABAC (etiquetas en políticas)	Sí
Credenciales temporales	Sí
Permisos de entidades principales	Sí
Roles de servicio	Sí
Roles vinculados al servicio	No

Para obtener una visión general de cómo funcionan Amazon GameLift y otros AWS servicios con la mayoría de las funciones de IAM, consulta AWSlos servicios que funcionan con IAM en la Guía del usuario de IAM.

Políticas basadas en la identidad de Amazon GameLift

Compatibilidad con las políticas basadas en identidades	Sí

Las políticas basadas en identidades son documentos de políticas de permisos JSON que puede adjuntar a una identidad, como un usuario, un grupo de usuarios o un rol de IAM. Estas políticas controlan qué acciones pueden realizar los usuarios y los roles, en qué recursos y en qué condiciones. Para obtener más información sobre cómo crear una política basada en identidad, consulte Creación de políticas de IAM en la Guía del usuario de IAM.

Con las políticas basadas en identidades de IAM, puede especificar las acciones y los recursos permitidos o denegados, así como las condiciones en las que se permiten o deniegan las acciones. No es posible especificar la entidad principal en una política basada en identidad porque se aplica al usuario o rol al que está adjunto. Para más información sobre los elementos que puede utilizar en

una política de JSON, consulte Referencia de los elementos de las políticas de JSON de IAM en la Guía del usuario de IAM.

Ejemplos de políticas basadas en identidad para Amazon GameLift

Para ver ejemplos de políticas de Amazon GameLift basadas en la identidad, consulta. Ejemplos de políticas basadas en identidad para Amazon GameLift

Políticas basadas en recursos en Amazon GameLift

Compatibilidad con las políticas basadas en recursos	No

Las políticas basadas en recursos son documentos de políticas JSON que se asocian a un recurso. Ejemplos de políticas basadas en recursos son las políticas de confianza de roles de IAM y las políticas de bucket de Amazon S3. En los servicios que admiten políticas basadas en recursos, los administradores de servicios pueden utilizarlos para controlar el acceso a un recurso específico. Para el recurso al que se asocia la política, la política define qué acciones puede realizar una entidad principal especificada en ese recurso y en qué condiciones. Debe especificar una entidad principal en una política en función de recursos. Las entidades principales pueden incluir cuentas, usuarios, roles, usuarios federados o servicios de Servicios de AWS.

Para habilitar el acceso entre cuentas, puede especificar toda una cuenta o entidades de IAM de otra cuenta como la entidad principal de una política en función de recursos. Añadir a una política en función de recursos una entidad principal entre cuentas es solo una parte del establecimiento de una relación de confianza. Cuando la entidad principal y el recurso se encuentran en Cuentas de AWS diferentes, un administrador de IAM de la cuenta de confianza también debe conceder a la entidad principal (usuario o rol) permiso para acceder al recurso. Para conceder el permiso, adjunte la entidad a una política basada en identidad. Sin embargo, si la política en función de recursos concede el acceso a una entidad principal de la misma cuenta, no es necesaria una política basada en identidad adicional. Para más información, consulte Cómo los roles de IAM difieren de las políticas basadas en recursos en la Guía del usuario de IAM.

Acciones políticas para Amazon GameLift

Admite acciones de políticas	Sí

Los administradores pueden utilizar las políticas JSON de AWS para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Action` de una política JSON describe las acciones que puede utilizar para conceder o denegar el acceso en una política. Las acciones de la política generalmente tienen el mismo nombre que la operación de API AWS asociada. Hay algunas excepciones, como acciones de solo permiso que no tienen una operación de API coincidente. También hay algunas operaciones que requieren varias acciones en una política. Estas acciones adicionales se denominan acciones dependientes.

Incluya acciones en una política para conceder permisos y así llevar a cabo la operación asociada.

Para ver una lista de GameLift las acciones de Amazon, consulta Acciones definidas por Amazon GameLift en la Referencia de autorización de servicio.

Las acciones políticas en Amazon GameLift usan el siguiente prefijo antes de la acción:

```
gamelift
```

Para especificar varias acciones en una única instrucción, sepárelas con comas.

```
"Action": [
      "gamelift:action1",
      "gamelift:action2"
         ]
```

Puede utilizar caracteres comodín (*) para especificar varias acciones. Por ejemplo, para especificar todas las acciones que comiencen con la palabra `Describe`, incluya la siguiente acción:

```
"Action": "gamelift:Describe*"
```

Para ver ejemplos de políticas de Amazon GameLift basadas en la identidad, consulta. Ejemplos de políticas basadas en identidad para Amazon GameLift

Recursos de políticas para Amazon GameLift

Admite recursos de políticas	Sí

Los administradores pueden utilizar las políticas JSON de AWS para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Resource` de la política JSON especifica el objeto u objetos a los que se aplica la acción. Las instrucciones deben contener un elemento `Resource` o `NotResource`. Como práctica recomendada, especifique un recurso utilizando el Nombre de recurso de Amazon (ARN). Puede hacerlo para acciones que admitan un tipo de recurso específico, conocido como permisos de nivel de recurso.

Para las acciones que no admiten permisos de nivel de recurso, como las operaciones de descripción, utilice un carácter comodín (*) para indicar que la instrucción se aplica a todos los recursos.

```
"Resource": "*"
```

Para obtener una lista de los tipos de GameLift recursos de Amazon y sus ARN, consulte Recursos definidos por Amazon GameLift en la Referencia de autorización de servicio. Para saber con qué acciones puede especificar el ARN de cada recurso, consulte Acciones definidas por Amazon. GameLift

Algunos GameLift recursos de Amazon tienen valores de ARN, lo que permite gestionar el acceso a los recursos mediante políticas de IAM. El recurso de GameLift flota de Amazon tiene un ARN con la siguiente sintaxis:

```
arn:${Partition}:gamelift:${Region}:${Account}:fleet/${FleetId}
```

Para obtener más información acerca del formato de los ARN, consulte Nombres de recursos de Amazon (ARN) en la Referencia general de AWS.

Por ejemplo, para especificar la flota `fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa` en su instrucción, utilice el siguiente ARN:

```
"Resource": "arn:aws:gamelift:us-west-2:123456789012:fleet/
fleet-2222bbbb-33cc-44dd-55ee-6666ffff77aa"
```

Para especificar todas las flotas que pertenecen a una cuenta específica, utilice el carácter comodín (*):

```
"Resource": "arn:aws:gamelift:us-west-2:123456789012:fleet/*"
```

Para ver ejemplos de políticas de Amazon GameLift basadas en la identidad, consulta. Ejemplos de políticas basadas en identidad para Amazon GameLift

Claves de condición de la política para Amazon GameLift

Admite claves de condición de políticas específicas del servicio	Sí

Los administradores pueden utilizar las políticas JSON de AWS para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Condition` (o bloque de `Condition`) permite especificar condiciones en las que entra en vigor una instrucción. El elemento `Condition` es opcional. Puede crear expresiones condicionales que utilicen operadores de condición, tales como igual o menor que, para que la condición de la política coincida con los valores de la solicitud.

Si especifica varios elementos de `Condition` en una instrucción o varias claves en un único elemento de `Condition`, AWS las evalúa mediante una operación lógica `AND`. Si especifica varios valores para una única clave de condición, AWS evalúa la condición con una operación `OR` lógica. Se deben cumplir todas las condiciones antes de que se concedan los permisos de la instrucción.

También puede utilizar variables de marcador de posición al especificar condiciones. Por ejemplo, puede conceder un permiso de usuario de IAM para acceder a un recurso solo si está etiquetado con su nombre de usuario de IAM. Para más información, consulte Elementos de la política de IAM: variables y etiquetas en la Guía del usuario de IAM.

AWS admite claves de condición globales y claves de condición específicas del servicio. Para ver todas las claves de condición globales de AWS, consulte Claves de contexto de condición globales de AWS en la Guía del usuario de IAM.

Para obtener una lista de las claves de GameLift estado de Amazon, consulta Claves de condición de Amazon GameLift en la Referencia de autorización de servicio. Para saber con qué acciones y recursos puede utilizar una clave de condición, consulte Acciones definidas por Amazon GameLift.

Para ver ejemplos de políticas de Amazon GameLift basadas en la identidad, consulta. Ejemplos de políticas basadas en identidad para Amazon GameLift

ACL en Amazon GameLift

Admite las ACL	No

Las listas de control de acceso (ACL) controlan qué entidades principales (miembros de cuentas, usuarios o roles) tienen permisos para acceder a un recurso. Las ACL son similares a las políticas basadas en recursos, aunque no utilizan el formato de documento de políticas JSON.

ABAC con Amazon GameLift

Admite ABAC (etiquetas en las políticas)	Sí

El control de acceso basado en atributos (ABAC) es una estrategia de autorización que define permisos en función de atributos. En AWS, estos atributos se denominan etiquetas. Puede adjuntar etiquetas a entidades de IAM (usuarios o roles) y a muchos recursos de AWS. El etiquetado de entidades y recursos es el primer paso de ABAC. A continuación, designa las políticas de ABAC para permitir operaciones cuando la etiqueta de la entidad principal coincida con la etiqueta del recurso al que se intenta acceder.

ABAC es útil en entornos que crecen con rapidez y ayuda en situaciones en las que la administración de las políticas resulta engorrosa.

Para controlar el acceso en función de etiquetas, debe proporcionar información de las etiquetas en el elemento de condición de una política utilizando las claves de condición `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` o `aws:TagKeys`.

Si un servicio admite las tres claves de condición para cada tipo de recurso, el valor es Sí para el servicio. Si un servicio admite las tres claves de condición solo para algunos tipos de recursos, el valor es Parcial.

Para obtener más información sobre ABAC, consulte ¿Qué es ABAC? en la Guía del usuario de IAM. Para ver un tutorial con los pasos para configurar ABAC, consulte Uso del control de acceso basado en atributos (ABAC) en la Guía del usuario de IAM.

Para obtener un ejemplo de política basada en identidad que limita el acceso a un recurso basado en las etiquetas de ese recurso, consulte Ver las GameLift flotas de Amazon en función de las etiquetas.

Uso de credenciales temporales con Amazon GameLift

Admite el uso de credenciales temporales	Sí

Algunos Servicios de AWS no funcionan cuando inicia sesión con credenciales temporales. Para obtener información adicional, incluida la información sobre qué servicios de Servicios de AWSfuncionan con credenciales temporales, consulte Servicios de Servicios de AWSque funcionan con IAM en la Guía del usuario de IAM.

Utilice credenciales temporales si inicia sesión en la AWS Management Console con cualquier método excepto un nombre de usuario y una contraseña. Por ejemplo, cuando accede a AWS utilizando el enlace de inicio de sesión único (SSO) de la empresa, ese proceso crea automáticamente credenciales temporales. También crea automáticamente credenciales temporales cuando inicia sesión en la consola como usuario y luego cambia de rol. Para más información sobrel cambio de roles, consulte Cambio a un rol (consola) en la Guía del usuario de IAM.

Puede crear credenciales temporales de forma manual mediante la AWS CLI o la API de AWS. A continuación, puede usar esas credenciales temporales para acceder a AWS. AWS recomienda generar credenciales temporales de forma dinámica en lugar de usar claves de acceso a largo plazo. Para más información, consulte Credenciales de seguridad temporales en IAM.

Permisos principales de servicios cruzados para Amazon GameLift

Admite Forward access sessions (FAS)	Sí

Cuando utiliza un usuario o un rol de IAM para llevar a cabo acciones en AWS, se lo considera una entidad principal. Cuando utiliza algunos servicios, es posible que realice una acción que desencadene otra acción en un servicio diferente. FAS utiliza los permisos de la entidad principal para llamar a un Servicio de AWS, combinados con el Servicio de AWS solicitante para realizar solicitudes a servicios posteriores. Las solicitudes de FAS solo se realizan cuando un servicio recibe una solicitud que requiere interacciones con otros Servicios de AWS o recursos para completarse. En este caso, debe tener permisos para realizar ambas acciones. Para obtener información sobre las políticas a la hora de realizar solicitudes de FAS, consulte Reenviar sesiones de acceso.

Funciones de servicio para Amazon GameLift

Compatible con roles de servicio	Sí

Un rol de servicio es un rol de IAM que asume un servicio para realizar acciones en su nombre. Un administrador de IAM puede crear, modificar y eliminar un rol de servicio desde IAM. Para más información, consulte Creación de un rol para delegar permisos a un Servicio de AWS en la Guía del usuario de IAM.

> **Warning**
>
> Cambiar los permisos de un rol de servicio podría interrumpir la GameLift funcionalidad de Amazon. Edita las funciones de servicio solo cuando Amazon te GameLift dé instrucciones para hacerlo.

Permita que sus servidores GameLift de juegos alojados en Amazon accedan a otros AWS recursos, como una AWS Lambda función o una base de datos de Amazon DynamoDB. Como los servidores de juegos se alojan en flotas GameLift gestionadas por Amazon, necesitas un rol de servicio que dé a Amazon un acceso GameLift limitado a tus otros AWS recursos. Para obtener más información, consulte Comunicación con otros recursos de AWS de sus flotas.

Funciones vinculadas a servicios para Amazon GameLift

Compatible con roles vinculados al servicio	No

Un rol vinculado al servicio es un tipo de rol de servicio que está vinculado a un Servicio de AWS. El servicio puede asumir el rol para realizar una acción en su nombre. Los roles vinculados a servicios aparecen en la Cuenta de AWS y son propiedad del servicio. Un administrador de IAM puede ver, pero no editar, los permisos de los roles vinculados a servicios.

Para obtener más información acerca de cómo crear o administrar roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM en la Guía de usuario de IAM. Busque un servicio en la tabla que incluya `Yes` en la columna Roles vinculados al servicio. Elija el vínculo Sí para ver la documentación sobre el rol vinculado a un servicio en cuestión.

Ejemplos de políticas basadas en identidad para Amazon GameLift

De forma predeterminada, los usuarios y los roles no tienen permiso para crear o modificar GameLift los recursos de Amazon. Tampoco pueden realizar tareas mediante la AWS Management Console, la AWS Command Line Interface (AWS CLI) o la API de AWS. Para conceder permiso a los usuarios para realizar acciones en los recursos que necesiten, un administrador de IAM puede crear políticas de IAM. A continuación, el administrador puede agregar las políticas de IAM a los roles y los usuarios pueden asumirlos.

Para obtener información acerca de cómo crear una política basada en identidades de IAM mediante el uso de estos documentos de políticas JSON de ejemplo, consulte Creación de políticas de IAM en la Guía del usuario de IAM.

Para obtener más información sobre las acciones y los tipos de recursos definidos por Amazon GameLift, incluido el formato de los ARN de cada uno de los tipos de recursos, consulte Acciones, recursos y claves de condición de Amazon GameLift en la Referencia de autorización de servicio.

Temas

Prácticas recomendadas sobre las políticas

Las políticas basadas en la identidad determinan si alguien puede crear, acceder o eliminar GameLift los recursos de Amazon de tu cuenta. Estas acciones pueden generar costes adicionales para su Cuenta de AWS. Siga estas directrices y recomendaciones al crear o editar políticas basadas en identidades:

- Comience con las políticas administradas de AWS y continúe con los permisos de privilegio mínimo: a fin de comenzar a conceder permisos a los usuarios y las cargas de trabajo, utilice las políticas administradas de AWS, que conceden permisos para muchos casos de uso comunes. Están disponibles en su Cuenta de AWS. Se recomienda definir políticas administradas por el

cliente de AWS específicas para los casos de uso a fin de reducir aún más los permisos. Con el fin de obtener más información, consulte las políticas administradas por AWS o las políticas administradas por AWS para funciones de trabajo en la Guía del usuario de IAM.

- Aplique permisos de privilegio mínimo: cuando establezca permisos con políticas de IAM, conceda solo los permisos necesarios para realizar una tarea. Para ello, debe definir las acciones que se pueden llevar a cabo en determinados recursos en condiciones específicas, también conocidos como permisos de privilegios mínimos. Con el fin de obtener más información sobre el uso de IAM para aplicar permisos, consulte Políticas y permisos en IAM en la Guía del usuario de IAM.
- Utilice condiciones en las políticas de IAM para restringir aún más el acceso: puede agregar una condición a sus políticas para limitar el acceso a las acciones y los recursos. Por ejemplo, puede escribir una condición de políticas para especificar que todas las solicitudes deben enviarse utilizando SSL. También puede usar condiciones para conceder acceso a acciones de servicios si se emplean a través de un Servicio de AWS determinado, como por ejemplo AWS CloudFormation. Para más información, consulte Elementos de la política JSON de IAM: condición en la Guía del usuario de IAM.
- Utilice el analizador de acceso de IAM para validar las políticas de IAM con el fin de garantizar la seguridad y funcionalidad de los permisos: el analizador de acceso de IAM valida políticas nuevas y existentes para que respeten el lenguaje (JSON) de las políticas de IAM y las prácticas recomendadas de IAM. El analizador de acceso de IAM proporciona más de 100 verificaciones de políticas y recomendaciones procesables para ayudar a crear políticas seguras y funcionales. Para más información, consulte la política de validación del Analizador de acceso de IAM en la Guía del usuario de IAM.
- Solicite la autenticación multifactor (MFA): si se encuentra en una situación en la que necesita usuarios raíz o de IAM en su Cuenta de AWS, active la MFA para mayor seguridad. Para solicitar la MFA cuando se invocan las operaciones de la API, agregue las condiciones de la MFA a sus políticas. Para más información, consulte Configuración de acceso a una API protegida por MFA en la Guía del usuario de IAM.

Para obtener más información sobre las prácticas recomendadas de IAM, consulte las Prácticas recomendadas de seguridad en IAM en la Guía del usuario de IAM.

Uso de la GameLift consola de Amazon

Para acceder a la GameLift consola de Amazon, debes tener un conjunto mínimo de permisos. Estos permisos deben permitirte enumerar y ver detalles sobre los GameLift recursos de Amazon que tienesCuenta de AWS. Si crea una política basada en identidades que sea más restrictiva que el

mínimo de permisos necesarios, la consola no funcionará del modo esperado para las entidades (usuarios o roles) que tengan esa política.

Para garantizar que esas entidades puedan seguir utilizando la GameLift consola de Amazon, añada permisos a los usuarios y grupos con la sintaxis de los siguientes ejemplos y enEjemplos de permisos de administrador. Para obtener más información, consulte Administrar los permisos de usuario para Amazon GameLift.

Los usuarios que trabajan con Amazon GameLift mediante AWS CLI operaciones de AWS API no requieren permisos mínimos de consola. En su lugar, puede limitar el acceso únicamente a las operaciones que el usuario debe realizar. Por ejemplo, un usuario jugador que actúa en nombre de los clientes de juegos necesita acceso para solicitar sesiones de juego, ubicar a los jugadores en los juegos y realizar otras tareas.

Para obtener información sobre los permisos necesarios para utilizar todas las funciones de la GameLift consola de Amazon, consulte la sintaxis de permisos para administradores enEjemplos de permisos de administrador.

Cómo permitir a los usuarios consultar sus propios permisos

En este ejemplo, se muestra cómo podría crear una política que permita a los usuarios de IAM ver las políticas administradas e insertadas que se asocian a la identidad de sus usuarios. Esta política incluye permisos para realizar esta acción en la consola o mediante programación con la AWS CLI o la API de AWS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
```

```
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Permitir acceso a los jugadores a las sesiones de juego

Para colocar a los jugadores en las sesiones de juego, los clientes de juegos y los servicios de backend necesitan permisos. Para ver ejemplos de políticas para estos escenarios, consulte Ejemplos de permisos de usuario de un jugador.

Permitir el acceso a una GameLift cola de Amazon

El siguiente ejemplo proporciona a un usuario acceso a GameLift colas de Amazon específicas.

Esta política otorga al usuario permisos para añadir, actualizar y eliminar destinos de colas con las siguientes acciones: `gamelift:UpdateGameSessionQueue`, `gamelift:DeleteGameSessionQueue` y `gamelift:DescribeGameSessionQueues`. Tal y como se muestra, esta política utiliza el elemento `Resource` para limitar el acceso a una sola cola: `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"
      }
   ]
}
```

Ver las GameLift flotas de Amazon en función de las etiquetas

Puedes usar las condiciones de tu política basada en la identidad para controlar el acceso a los GameLift recursos de Amazon en función de las etiquetas. Este ejemplo muestra cómo puede crear una política que permita ver una flota si la etiqueta `Owner` coincide con el nombre de usuario del usuario. Esta política también proporciona los permisos necesarios para completar esta operación en la consola.

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

Acceso a un archivo de compilación de un juego en Amazon S3

Tras integrar el servidor de juegos con Amazon GameLift, sube los archivos de compilación a Amazon S3. Para GameLift que Amazon pueda acceder a los archivos de compilación, usa la siguiente política.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": "arn:aws:s3:::bucket-name/object-name"
        }
    ]
}
```

Para obtener más información sobre la carga de archivos de GameLift juegos de Amazon, consultaCarga de una compilación del servidor de juegos personalizada en Amazon GameLift.

Solución de problemas de GameLift identidad y acceso a Amazon

Usa la siguiente información para ayudarte a diagnosticar y solucionar problemas comunes que podrías encontrar al trabajar con Amazon GameLift e AWS Identity and Access Management (IAM).

Temas

- No estoy autorizado a realizar ninguna acción en Amazon GameLift
- No estoy autorizado a realizar lo siguiente: PassRole
- Quiero permitir que personas ajenas a mí accedan Cuenta de AWS a mis GameLift recursos de Amazon

No estoy autorizado a realizar ninguna acción en Amazon GameLift

Si la AWS Management Console le indica que no está autorizado para llevar a cabo una acción, póngase en contacto con su administrador de cuentas de AWS para recibir ayuda. El administrador es la persona que le proporcionó las credenciales de inicio de sesión.

En el siguiente ejemplo, el error se produce cuando el usuario de IAM, `mateojackson`, intenta utilizar la consola para ver detalles sobre una cola, pero no tiene permisos de `gamelift:DescribeGameSessionQueues`.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 gamelift:DescribeGameSessionQueues on resource: examplequeue123
```

En este caso, Mateo pide a su administrador que actualice sus políticas para poder tener acceso de lectura al recurso `examplequeue123` mediante la acción `gamelift:DescribeGameSessionQueues`.

No estoy autorizado a realizar lo siguiente: PassRole

Si recibes un error que indica que no estás autorizado a realizar la `iam:PassRole` acción, debes actualizar tus políticas para que puedas transferir una función a Amazon GameLift.

Algunos Servicios de AWS le permiten transferir un rol existente a dicho servicio en lugar de crear un nuevo rol de servicio o uno vinculado al servicio. Para ello, debe tener permisos para transferir el rol al servicio.

El siguiente ejemplo de error se produce cuando un usuario de IAM denominado `marymajor` intenta utilizar la consola para realizar una acción en Amazon GameLift. Sin embargo, la acción requiere que el servicio cuente con permisos que concede un rol de servicio. Mary no tiene permisos para transferir el rol al servicio.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

En este caso, las políticas de Mary se deben actualizar para permitirle realizar la acción `iam:PassRole`.

Si necesita ayuda, póngase en contacto con su administrador de AWS. El administrador es la persona que le proporcionó las credenciales de inicio de sesión.

Quiero permitir que personas ajenas a mí accedan Cuenta de AWS a mis GameLift recursos de Amazon

Puede crear un rol que los usuarios de otras cuentas o las personas externas a la organización puedan utilizar para acceder a sus recursos. Puede especificar una persona de confianza para que asuma el rol. En el caso de los servicios que admitan las políticas basadas en recursos o las listas de control de acceso (ACL), puede utilizar dichas políticas para conceder a las personas acceso a sus recursos.

Para más información, consulte lo siguiente:

- Para saber si Amazon GameLift admite estas funciones, consultaCómo GameLift funciona Amazon con IAM.
- Para obtener información acerca de cómo proporcionar acceso a los recursos de las Cuentas de AWS de su propiedad, consulte Proporcionar acceso a un usuario de IAM a otra Cuenta de AWS de la que es propietario en la Guía del usuario de IAM.
- Para obtener información acerca de cómo proporcionar acceso a los recursos a Cuentas de AWS de terceros, consulte Proporcionar acceso a Cuentas de AWS que son propiedad de terceros en la Guía del usuario de IAM.
- Para obtener información sobre cómo proporcionar acceso mediante una federación de identidades, consulte Proporcionar acceso a usuarios autenticados externamente (federación de identidades) en la Guía del usuario de IAM.
- Para obtener información sobre la diferencia entre los roles y las políticas basadas en recursos para el acceso entre cuentas, consulte Cómo los roles de IAM difieren de las políticas basadas en recursos en la Guía del usuario de IAM.

Registro y supervisión con Amazon GameLift

La supervisión es una parte importante a la hora de mantener la fiabilidad, la disponibilidad y el rendimiento de Amazon GameLift y de otras soluciones de AWS. Debe recopilar datos de monitorización de todas las partes de su solución de AWS para que le resulte más sencillo depurar un error que se produce en distintas partes del código, en caso de que ocurra.

AWS y Amazon GameLift proporcionan varias herramientas para supervisar sus recursos de alojamiento de juegos y responder a posibles incidentes.

Alarmas de Amazon CloudWatch

Con las alarmas de Amazon CloudWatch, puede ver una métrica determinada durante el periodo especificado. Si la métrica supera un límite determinado, se envía una notificación a un tema de Amazon SNS o a una política de AWS Auto Scaling. Las alarmas de CloudWatch se activan al cambiar su estado, no por estar en un estado determinado, y se mantienen durante un número de períodos especificado. Para obtener más información, consulte Supervisión de Amazon GameLift con Amazon CloudWatch.

Registros de AWS CloudTrail

CloudTrail proporciona un registro de las acciones que realiza un usuario, rol o servicio de AWS en Amazon GameLift. Mediante la información recopilada por CloudTrail, puede determinar la solicitud que se realizó a Amazon GameLift, la dirección IP de origen desde la que se realizó la solicitud, quién realizó la solicitud, cuándo se realizó y otros detalles adicionales. Para obtener más información, consulte Registro de llamadas a la API de Amazon GameLift con AWS CloudTrail.

Validación de conformidad para Amazon GameLift

Amazon no GameLift está dentro del ámbito de ningún programa de AWS conformidad.

Para saber si un programa de cumplimiento Servicio de AWS está incluido Servicios de AWS en el ámbito de aplicación de un programa de cumplimiento específico, consulta el de cumplimiento y selecciona el programa de cumplimiento que te interese. Para obtener información general, consulte Programas de AWS cumplimiento > Programas AWS .

Puede descargar informes de auditoría de terceros utilizando AWS Artifact. Para obtener más información, consulte Descarga de informes en AWS Artifact .

Su responsabilidad de cumplimiento al Servicios de AWS utilizarlos viene determinada por la confidencialidad de sus datos, los objetivos de cumplimiento de su empresa y las leyes y reglamentos aplicables. AWS proporciona los siguientes recursos para ayudar con el cumplimiento:

- Guías de inicio rápido sobre seguridad y cumplimiento: estas guías de implementación analizan las consideraciones arquitectónicas y proporcionan los pasos para implementar entornos básicos centrados en AWS la seguridad y el cumplimiento.
- Diseño de arquitectura para garantizar la seguridad y el cumplimiento de la HIPAA en Amazon Web Services: en este documento técnico se describe cómo pueden utilizar AWS las empresas para crear aplicaciones aptas para la HIPAA.

> **Note**
>
> No Servicios de AWS todas cumplen con los requisitos de la HIPAA. Para más información, consulte la Referencia de servicios compatibles con HIPAA.

- AWS Recursos de de cumplimiento: esta colección de libros de trabajo y guías puede aplicarse a su industria y ubicación.
- AWS Guías de cumplimiento para clientes: comprenda el modelo de responsabilidad compartida desde el punto de vista del cumplimiento. Las guías resumen las mejores prácticas para garantizar la seguridad Servicios de AWS y orientan los controles de seguridad en varios marcos (incluidos el Instituto Nacional de Estándares y Tecnología (NIST), el Consejo de Normas de Seguridad del Sector de Tarjetas de Pago (PCI) y la Organización Internacional de Normalización (ISO)).
- Evaluación de los recursos con reglas en la guía para AWS Config desarrolladores: el AWS Config servicio evalúa en qué medida las configuraciones de los recursos cumplen con las prácticas internas, las directrices del sector y las normas.
- AWS Security Hub— Esto Servicio de AWS proporciona una visión completa del estado de su seguridad interior AWS. Security Hub utiliza controles de seguridad para evaluar sus recursos de AWS y comprobar su cumplimiento con los estándares y las prácticas recomendadas del sector de la seguridad. Para obtener una lista de los servicios y controles compatibles, consulte la Referencia de controles de Security Hub.
- Amazon GuardDuty: Servicio de AWS detecta posibles amenazas para sus cargas de trabajo Cuentas de AWS, contenedores y datos mediante la supervisión de su entorno para detectar actividades sospechosas y maliciosas. GuardDuty puede ayudarlo a cumplir con varios requisitos de conformidad, como el PCI DSS, al cumplir con los requisitos de detección de intrusiones exigidos por ciertos marcos de cumplimiento.
- AWS Audit Manager— Esto le Servicio de AWS ayuda a auditar continuamente su AWS uso para simplificar la gestión del riesgo y el cumplimiento de las normativas y los estándares del sector.

Resiliencia en Amazon GameLift

Si utiliza Amazon GameLift FleetIQ como función independiente con Amazon EC2, consulte Seguridad en Amazon EC2 en la Guía del usuario de Amazon EC2.

La infraestructura AWS global se basa en regiones y zonas de disponibilidad. AWS AWS Las regiones proporcionan varias zonas de disponibilidad aisladas y separadas físicamente, que están

conectadas mediante redes de baja latencia, alto rendimiento y alta redundancia. Con las zonas de disponibilidad, puede diseñar y utilizar aplicaciones y bases de datos que realizan una conmutación por error automática entre las zonas sin interrupciones. Las zonas de disponibilidad tienen una mayor disponibilidad, tolerancia a errores y escalabilidad que las infraestructuras tradicionales de uno o varios centros de datos.

Para obtener más información sobre AWS las regiones y las zonas de disponibilidad, consulte la infraestructura global.AWS

Además de la infraestructura AWS global, Amazon GameLift ofrece las siguientes funciones para ayudarlo a satisfacer sus necesidades de resiliencia de datos:

- Colas multirregionales: las colas de sesiones de GameLift juego de Amazon se utilizan para colocar nuevas sesiones de juego con los recursos de alojamiento disponibles. Las colas que abarcan varias regiones pueden redirigir las ubicaciones de sesiones de juego en caso de que se produzca una interrupción regional. Para obtener más información y conocer las prácticas recomendadas sobre la creación de colas de sesiones de juego, consulte Diseño de colas de sesiones de juego.
- Escalado automático de la capacidad: mantenga el estado y la disponibilidad de sus recursos de alojamiento mediante las herramientas de GameLift escalado de Amazon. Estas herramientas ofrecen una amplia gama de opciones que le permiten ajustar la capacidad de la flota para adaptarla a las necesidades del juego y de los jugadores. Para obtener más información sobre el escalado, consulte Escalación de la capacidad de alojamiento de Amazon GameLift.
- Distribución entre instancias: Amazon GameLift distribuye el tráfico entrante entre varias instancias, según el tamaño de la flota. Como práctica recomendada, los juegos en producción deben tener varias instancias para mantener la disponibilidad en caso de que una instancia deje de estar en buen estado o no responda.
- Almacenamiento en Amazon S3: las compilaciones y los scripts de los servidores de juegos que se cargan en Amazon se GameLift almacenan en Amazon S3 mediante la clase de almacenamiento estándar, que utiliza múltiples replicaciones de centros de datos para aumentar la resiliencia. Los registros de sesiones de juego también se almacenan en Amazon S3 con la clase de almacenamiento estándar.

Seguridad de la infraestructura en Amazon GameLift

Si utiliza Amazon GameLift FleetIQ como función independiente con Amazon EC2, consulte Seguridad en Amazon EC2 en la Guía del usuario de Amazon EC2.

Como servicio gestionado, Amazon GameLift está protegido por los procedimientos de seguridad de la red AWS global que se describen en el documento técnico Amazon Web Services: descripción general de los procesos de seguridad.

Utilizas las llamadas a la API AWS publicadas para acceder a Amazon GameLift a través de la red. Los clientes deben ser compatibles con Transport Layer Security (TLS) 1.2 o una versión posterior. Recomendamos TLS 1.3 o una versión posterior. Los clientes también deben ser compatibles con conjuntos de cifrado con confidencialidad directa total (PFS) tales como Ephemeral Diffie-Hellman (DHE) o Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). La mayoría de los sistemas modernos como Java 7 y posteriores son compatibles con estos modos.

Además, las solicitudes deben estar firmadas mediante un ID de clave de acceso y una clave de acceso secreta que esté asociada a una entidad de seguridad de IAM principal. También puede utilizar AWS Security Token Service (AWS STS) para generar credenciales de seguridad temporales para firmar solicitudes.

El GameLift servicio de Amazon coloca todas las flotas en las nubes privadas virtuales (VPC) de Amazon para que cada flota se encuentre en un área aislada lógicamente en la nube. AWS Puedes usar GameLift las políticas de Amazon para controlar el acceso desde puntos de enlace de VPC específicos o VPC específicas. De hecho, esto aísla el acceso a la red a un GameLift recurso de Amazon determinado únicamente de la VPC específica de la red. AWS Al crear una flota, se especifica un intervalo de números de puerto y direcciones IP. Estos intervalos limitan el modo en que el tráfico entrante puede obtener acceso a los servidores de juegos alojados en la VPC de una flota. Utilice las prácticas recomendadas de seguridad estándar al elegir la configuración de acceso a la flota.

Análisis de configuración y vulnerabilidad en Amazon GameLift

Si utiliza Amazon GameLift FleetIQ como función independiente con Amazon EC2, consulte Seguridad en Amazon EC2 en la Guía del usuario de Amazon EC2.

La configuración y los controles de TI son una responsabilidad compartida entre AWS y usted, nuestro cliente. Para obtener más información, consulte el modelo de responsabilidad compartida. AWS AWS gestiona las tareas de seguridad básicas, como la aplicación de parches al sistema operativo (SO) huésped y a las bases de datos, la configuración del firewall y la recuperación ante desastres. Estos procedimientos han sido revisados y certificados por los terceros pertinentes. Para obtener más información, consulte el recurso siguiente: Amazon Web Services: Información general de procesos de seguridad (documento técnico).

Las siguientes prácticas recomendadas de seguridad también abordan la configuración y el análisis de vulnerabilidades en Amazon GameLift:

- Los clientes son responsables de la administración del software que se implementa en las GameLift instancias de Amazon para el alojamiento de juegos. En concreto:
  - El software de las aplicaciones del servidor de juegos que proporciona el cliente requiere un mantenimiento, que incluye actualizaciones y parches de seguridad. Para actualizar el software del servidor de juegos, sube una nueva versión a Amazon GameLift, crea una nueva flota y redirige el tráfico a la nueva flota.
  - La imagen de Amazon Machine (AMI) base, que incluye el sistema operativo, solo se actualiza cuando se crea una flota. Para aplicar parches, actualizar y proteger el sistema operativo y otras aplicaciones que forman parte de la AMI, recicle las flotas periódicamente, independientemente de las actualizaciones del servidor de juegos.
- Los clientes deberían considerar la posibilidad de actualizar sus juegos con regularidad con las versiones más recientes del AWS SDK, incluidos el SDK, el Amazon GameLift Server SDK y el Amazon GameLift Client SDK for Realtime Servers.

Mejores prácticas de seguridad para Amazon GameLift

Si utiliza Amazon GameLift FleetIQ como función independiente con Amazon EC2, consulte Seguridad en Amazon EC2 en la Guía del usuario de Amazon EC2.

Amazon GameLift proporciona una serie de características de seguridad que debes tener en cuenta a la hora de desarrollar e implementar tus propias políticas de seguridad. Las siguientes prácticas recomendadas son directrices generales y no suponen una solución de seguridad completa. Puesto que es posible que estas prácticas recomendadas no sean adecuadas o suficientes para el entorno, considérelas como consideraciones útiles en lugar de como normas.

No abra puertos a Internet

Recomendamos encarecidamente no abrir puertos a Internet porque hacerlo supone un riesgo para la seguridad. Por ejemplo, si solías UpdateFleetPortSettingsabrir un puerto de escritorio remoto como este:

```
{
  "FleetId": "<fleet identifier>",
  "InboundPermissionAuthorizations": [
```

```
      {
        "FromPort": 3389,
        "IpRange": "0.0.0.0/0",
        "Protocol": "RDP",
        "ToPort": 3389
      }
  ]
}
```

entonces está permitiendo que cualquier usuario de Internet acceda a la instancia.

En su lugar, abre el puerto con una dirección IP específica o un rango de direcciones. Por ejemplo, así:

```
{
  "FleetId": "<fleet identifier>",
  "InboundPermissionAuthorizations": [
      {
        "FromPort": 3389,
        "IpRange": "54.186.139.221/32",
        "Protocol": "TCP",
        "ToPort": 3389
      }
  ]
}
```

Más información

Para obtener más información sobre cómo puedes hacer que tu uso de Amazon sea GameLift más seguro, consulta el pilar AWS Well-Architected Tool de seguridad. .

Guías de referencia de Amazon GameLift

Esta sección contiene la documentación de referencia para utilizar Amazon GameLift.

Temas

Referencia de la API del servicio de Amazon GameLift (SDK de AWS)

En este tema se proporciona una lista de operaciones de la API basada en tareas para su uso con las soluciones de alojamiento administrado de Amazon GameLift, incluido el alojamiento de servidores de juegos personalizados y servidores de Realtime. Estas operaciones se incluyen en el SDK de AWS, en el espacio de nombres de `aws.gamelift`. Descargue el SDK de AWS oconsulte la documentación de referencia de la API de Amazon GameLift.

La API incluye dos conjuntos de operaciones para el alojamiento administrado de juegos:

- Configuración y administración de los recursos de alojamiento de Amazon GameLift
- Inicio de sesiones de juego y unión de los jugadores

La API del servicio de Amazon GameLift también contiene operaciones para su uso con otras herramientas y soluciones de Amazon GameLift. Para obtener una lista de API de FleetIQ, consulte Acciones de la API de FleetIQ. Para obtener una lista de las API de FlexMatch para el emparejamiento, consulte las acciones de la API de FlexMatch.

Configuración y administración de los recursos de alojamiento de Amazon GameLift

Llame a estas operaciones para configurar los recursos de alojamiento para sus servidores de juegos, escalar la capacidad para satisfacer la demanda de jugadores y acceder a métricas de

rendimiento y utilización, entre otras cosas. Estas operaciones de la API se utilizan con servidores de juegos alojados en Amazon GameLift, incluido Realtime Servers. Puede utilizar la consola de Amazon GameLift (AWS Command Line Interface) para la mayoría de las tareas de administración de recursos o puede realizar llamadas al servicio mediante la herramienta (AWS CLI) o el SDK de AWS.

Preparación de servidores de juegos para la implementación

Cargue y configure el código del servidor de juegos del juego para prepararlo para su implementación y lanzamiento en los recursos de alojamiento.

Administración de compilaciones de servidores de juegos personalizados

- upload-build: permite cargar archivos de compilación desde una ruta local y crear un nuevo recurso de compilación de Amazon GameLift. Esta operación, disponible solo como un comando AWS CLI, es el método más común para cargar compilaciones de servidores de juegos.
- CreateBuild: permite crear una nueva compilación de juego utilizando archivos almacenados en un bucket de Amazon S3.
- ListBuilds: permite obtener una lista de todas las compilaciones cargadas en una región de Amazon GameLift.
- DescribeBuild: permite recuperar la información asociada a una compilación.
- UpdateBuild: permite cambiar los metadatos de la compilación, incluidos el nombre y la versión de la compilación.
- DeleteBuild: permite eliminar una compilación de Amazon GameLift.

Administración de los scripts de configuración de Realtime Servers

- CreateScript: permite cargar archivos JavaScript y crear un nuevo recurso de script de Amazon GameLift.
- ListScripts: permite obtener una lista de todos los scripts cargados en una región de Amazon GameLift.
- DescribeScript: permite recuperar la información asociada a un script de Realtime.
- Updatescript: permite cambiar los metadatos del script y cargar el contenido del script revisado.
- DeleteScript: permite eliminar un script de Realtime de Amazon GameLift.

Configuración de los recursos informáticos para el alojamiento

Configure los recursos de alojamiento e impleméntelos con la compilación del servidor de juegos o el script de configuración de Realtime.

Creación y administración de flotas

- CreateFleet: permite configurar e implementar una nueva flota de recursos informáticos de Amazon GameLift para ejecutar sus servidores de juego. Una vez implementados, los servidores de juegos se lanzan automáticamente según estén configurados y preparados para alojar sesiones de juego.
- ListFleets: permite obtener una lista de todas las flotas de una región de Amazon GameLift.
- DeleteFleet: permite finalizar una flota que ya no ejecute servidores de juegos ni aloje jugadores.
- Visualización/actualización de las ubicaciones de la flota
  - CreateFleetLocations: permite añadir ubicaciones remotas a una flota existente que admite varias ubicaciones.
  - DescribeFleetLocationAttributes: permite obtener una lista de todas las ubicaciones remotas de una flota y ver el estado actual de cada ubicación.
  - DeleteFleetLocations: permite eliminar las ubicaciones remotas de una flota que admite varias ubicaciones.
- Consulte o actualice las configuraciones de la flota.
  - DescribeFleetAttributes/UpdateFleetAttributes: permite ver o cambiar los metadatos y la configuración de una flota para la protección de las sesiones de juego y los límites de creación de recursos.
  - DescribeFleetPortSettings/UpdateFleetPortSettings: permite ver o cambiar los permisos de entrada (rangos de configuración de puertos y direcciones IP) permitidos para una flota.
  - DescriberUntimeConfiguration/UpdaterUntimeConfiguration: permite ver o cambiar los procesos de servidor (y la cantidad) para ejecutar en cada instancia de una flota.

Administración de la capacidad de la flota

- DescribeEC2InstanceLimits: permite recuperar el número máximo de instancias permitido para la cuenta de AWS actual y el nivel de uso actual.
- DescribeFleetCapacity: permite recuperar la configuración de la capacidad actual de la región de origen de una flota.

- DescribeFleetLocationCapacity: permite recuperar la configuración de capacidad actual de cada ubicación de una flota con varias ubicaciones.
- UpdatefleetCapacity: permite ajusta manualmente la configuración de la capacidad de una flota.
- Configuración de escalado automático:
  - PutsCalingPolicy: permite activar el escalado automático basado en objetivos, crear una política de escalado automático personalizada o actualizar una política existente.
  - DescribEscalingPolicies: permite recuperar una política de escalado automático existente.
  - DeletEscalingPolicy: permite eliminar una política de escalado automático y evitar que afecte a la capacidad de la flota.
  - StartFleetActions: permite reiniciar políticas de escalado automático de una flota.
  - StopFleetActions: permite suspender las políticas de escalado automático de una flota.

Monitorice la actividad de la flota.

- DescribeFleetUtilization: permite recuperar estadísticas sobre la cantidad de procesos del servidor, sesiones de juego y jugadores que están activos actualmente en una flota.
- DescribeFleetLocationUtilization: permite recuperar las estadísticas de utilización de cada ubicación en una flota con varias ubicaciones.
- DescribeFleetEvents: permite ver los eventos registrados de una flota durante un periodo específico.
- DescribeGamessions: permite recuperar los metadatos de las sesiones de juego, incluido el tiempo de ejecución de un juego y el número actual de jugadores.

Configurar colas para una ubicación óptima de la sesión de juego

Configure colas de varias flotas y regiones para colocar las sesiones de juego con los mejores recursos de alojamiento disponibles en cuanto a costo, latencia y resiliencia.

- CreateGamessionQueue: permite crear una cola para usarla al procesar las solicitudes de ubicación de las sesiones de juego.
- DescribeGamessionQueues: permite recuperar las colas de sesiones de juego definidas en una región de Amazon GameLift.
- UpdateGamessionQueue: permite cambiar la configuración de una cola de sesión de juego.
- DeleteGamessionQueue: permite eliminar una cola de sesión de juego de la región.

Administrar alias

Utilice alias para representar sus flotas o crear un destino alternativo de terminal. Los alias son útiles cuando se pasa la actividad del juego de una flota a otra, como durante las actualizaciones de compilación del servidor de juegos.

- CreateAlias: permite definir un nuevo alias y, si lo desea, asignarlo a una flota.
- ListAliases: permite obtener todos los alias de flota definidos en una región de Amazon GameLift.
- describeAlias: permite recuperar información sobre un alias existente.
- UpdateAlias: permite cambiar la configuración de un alias, por ejemplo, redirigirlo de una flota a otra.
- DeleteAlias: permite eliminar un alias de la región.
- resolveAlias: permite obtener el ID de la flota al que apunta un alias específico.

Acceder a instancias de alojamiento

Consulte información sobre las distintas instancias de una flota o solicite el acceso remoto a la instancia de una flota especificada para solucionar problemas.

- DescribeInstances: permite obtener información sobre cada instancia de una flota, incluidos el ID de la instancia, la dirección IP, la ubicación y el estado.
- GetInstanceAccess: permite solicitar las credenciales de acceso necesarias para conectarse de forma remota a una instancia específica de una flota.

Configurar las interconexiones de VPC

Cree y administre conexiones de emparejamiento de VPC entre sus recursos de alojamiento de Amazon GameLift y otros recursos de AWS.

- CreateVpcPeeringAuthorization: permite autorizar una conexión de emparejamiento a una de sus VPC.
- DescribevpcPeeringAuthorization: permite recuperar autorizaciones de conexión de emparejamiento válidas.
- DeleteVpcPeeringAuthorization: permite eliminar una autorización de conexión de emparejamiento.
- CreateVPCpeeringConnection: permite establecer una conexión de emparejamiento entre la VPC de una flota de Amazon GameLift y una de sus VPC.

- DescribeVpcPeeringConnections: permite recuperar información sobre las conexiones de emparejamiento de VPC activas o pendientes con una flota de Amazon GameLift.
- DeleteVpcPeeringConnection: permite eliminar una conexión de emparejamiento de VPC con una flota de Amazon GameLift.

Inicio de sesiones de juego y unión de los jugadores

Utilice estas operaciones desde el servicio de cliente de juegos para iniciar nuevas sesiones de juego, obtener información sobre las sesiones de juego existentes y unir a los jugadores a las sesiones de juego. Estas operaciones son para los servidores de juegos personalizados alojados en Amazon GameLift. Si utiliza Realtime Servers, administra las sesiones de juego mediante el Referencia de la API de cliente de Realtime Servers (C#).

- Inicie nuevas sesiones de juego para uno o varios jugadores.
  - StartGamessionPlacement: permite solicitar a Amazon GameLift que busque los mejores recursos de alojamiento disponibles e iniciar una nueva sesión de juego. Este es el método preferido para crear nuevas sesiones de juego. Se basa en las colas de sesiones de juego para realizar un seguimiento de la disponibilidad del alojamiento en varias regiones y utilizar los algoritmos de FleetIQ para priorizar las ubicaciones en función de la latencia de los jugadores, el costo del alojamiento, la ubicación, etc.
  - DescribeGamessionPlacement: permite obtener los detalles y el estado de una solicitud de ubicación.
  - StopGamessionPlacement: permite cancelar una solicitud de ubicación.
  - CreateGamesession: permite iniciar una nueva sesión de juego vacía en una ubicación de flota específica. Esta operación le proporciona un mayor control sobre dónde iniciar la sesión de juego en lugar de utilizar FleetIQ para evaluar las opciones de ubicación. Debe añadir jugadores a la nueva sesión de juego en un paso aparte.
- Coloque a los jugadores en sesiones de juego existentes. Busque sesiones de juego en ejecución con ranuras de jugador disponibles y resérvelas para nuevos jugadores.
  - CreatePlayerSession: permite reservar una ranura libre para que un jugador se una a una sesión de juego.
  - CreatePlayerSessions: permite reservar ranuras abiertas para que varios jugadores se unan a una sesión de juego.
- Trabaje con datos de sesiones de juego y de jugador. Administración de información sobre sesiones de juego y sesiones de jugador

- SearchGamesessions: permite solicitar una lista de las sesiones de juego activas en función de un conjunto de criterios de búsqueda.
- DescribeGamessions: permite recuperar los metadatos de las sesiones de juego específicas, incluido el tiempo de activación y el número actual de jugadores.
- DescribeGamessionDetails: permite recuperar los metadatos, incluida la configuración de protección de la sesión de juego de una o más sesiones de juego.
- DescribePlayerSessions: permite obtener información sobre la actividad de los jugadores, incluidos el estado, el tiempo de juego y los datos del jugador.
- UpdateGameSession: permite cambiar la configuración de la sesión de juego, como el número máximo de jugadores y la política de unión.
- GetGamessionLogurl: permite obtener la ubicación de los registros almacenados para una sesión de juego.

Referencia de Servidores en tiempo real de Amazon GameLift

Esta sección contiene la documentación de referencia del SDK de Servidores en tiempo real de Amazon GameLift Incluye la API de cliente de Realtime, así como instrucciones para configurar el script de Realtime Servers.

Temas

- Referencia de la API de cliente de Realtime Servers (C#)
- Referencia de scripts de Servidores en tiempo real de Amazon GameLift

Referencia de la API de cliente de Realtime Servers (C#)

Utilice la API de cliente de Realtime para preparar sus clientes de juego multijugador para utilizarlos con Servidores en tiempo real de Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Preparación del servidor de Realtime. La API de cliente contiene un conjunto de llamadas a la API síncronas y devoluciones de llamadas asíncronas que permiten a un cliente del juego conectarse a un servidor de Realtime e intercambiar mensajes y datos con otros clientes del juego a través del servidor.

Esta API se define en las siguientes bibliotecas:

Client.cs

- Acciones síncronas
- Devoluciones de llamadas asíncronas
- Tipos de datos

Para configurar la API de cliente de Realtime, realice el siguiente procedimiento:

1. Descargue el SDK de cliente de Realtime de Amazon GameLift.
2. Compile las bibliotecas del SDK de C#. Localice el archivo de solución `GameLiftRealtimeClientSdkNet45.sln`. Consulte el archivo `README.md` para el SDK del servidor de C# para conocer los requisitos mínimos y las opciones de compilación adicionales. En un IDE, cargue el archivo de la solución. Para generar las bibliotecas del SDK, restaure los paquetes NuGet y compile la solución.
3. Añada las bibliotecas de cliente de Realtime a su proyecto de cliente de juegos.

Referencia de la API de cliente de Realtime Servers (C#): Acciones

Esta referencia de la API de cliente de Realtime de C# puede ayudarle a preparar el juego multijugador para utilizarlo con servidores de Realtime Servers implementados en flotas de Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Preparación del servidor de Realtime.

- Acciones síncronas
- Devoluciones de llamadas asíncronas
- Tipos de datos

Client()

Inicializa un nuevo cliente para comunicarse con el servidor Realtime e identifica el tipo de conexión que se va a utilizar.

Sintaxis

```
public Client(ClientConfiguration configuration)
```

Parámetros

clientConfiguration

Detalles de configuración que especifican el tipo de conexión cliente/servidor. Puede optar por llamar a Client() sin este parámetro; sin embargo, este enfoque da como resultado una conexión no segura de forma predeterminada.

Escriba: ClientConfiguration

Obligatorio: no

Valor devuelto

Devuelve una instancia del cliente de Realtime para comunicarse con el servidor Realtime.

Connect()

Solicita una conexión a un proceso del servidor que aloja una sesión de juego.

Sintaxis

```
public ConnectionStatus Connect(string endpoint, int remoteTcpPort, int listenPort,
 ConnectionToken token)
```

Parámetros

punto de conexión

Nombre de DNS o dirección IP de la sesión de juego a la que conectarse. El punto de conexión se especifica en un objeto `GameSession`, que se devuelve en respuesta a una llamada de cliente a las acciones de la API de Amazon GameLift del SDK de AWS StartGameSessionPlacement, CreateGameSession o DescribeGameSessions.

> **Note**
>
> Si el servidor Realtime se ejecuta en una flota con un certificado TLS, debe utilizar el nombre de DNS.

Tipo: String

Obligatorio: sí

remoteTcpPort

Número de puerto para la conexión TCP asignada a la sesión de juego. Esta información se especifica en un objeto `GameSession`, que se devuelve en respuesta a una solicitud StartGameSessionPlacement CreateGameSession o DescribeGameSession.

Tipo: entero

Valores válidos: 1900 - 2000.

Obligatorio: sí

listenPort

Número de puerto al que escucha el cliente del juego para mensajes enviados con el canal UDP.

Tipo: entero

Valores válidos: 33400 - 33500.

Obligatorio: sí

token

Información opcional que identifica al cliente de juego solicitante del proceso del servidor.

Escriba: ConnectionToken

Obligatorio: sí

Valor devuelto

Devuelve un valor enum ConnectionStatus que indica el estado de conexión del cliente.

Disconnect()

Cuando se está conectado a una sesión del juego, desconecta el cliente del juego de la sesión del juego.

Sintaxis

```
public void Disconnect()
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Este método no devuelve nada.

NewMessage()

Crea un nuevo objeto de mensaje con un código de operación especificado. Tras devolver un objeto de mensaje, complete el contenido del mensaje especificando un objetivo, actualizando el método de entrega y añadiendo una carga de datos según sea necesario. Tras completar este proceso, envíe el mensaje con `SendMessage()`.

Sintaxis

```
public RTMessage NewMessage(int opCode)
```

Parámetros

opCode

Código de operación definido por el desarrollador que identifica un evento o acción del juego, como un movimiento del jugador o una notificación del servidor.

Tipo: entero

Obligatorio: sí

Valor devuelto

Devuelve un objeto RTMessage que contiene el código de operación especificado y el método de entrega predeterminado. El parámetro de intento de entrega está configurado como `FAST` de forma predeterminada.

SendMessage()

Envía un mensaje a un jugador o grupo mediante el método de entrega especificado.

Sintaxis

```
public void SendMessage(RTMessage message)
```

Parámetros

message

Objeto de mensaje que especifica el destinatario objetivo, el método de entrega y el contenido del mensaje.

Escriba: RTMessage

Obligatorio: sí

Valor devuelto

Este método no devuelve nada.

JoinGroup()

Añade el jugador a la pertenencia a un grupo especificado. Los grupos pueden contener a cualquiera de los jugadores conectados al juego. Tras unirse, el jugador recibe todos los mensajes futuros enviados al grupo y puede enviar los mensajes a la totalidad del grupo.

Sintaxis

```
public void JoinGroup(int targetGroup)
```

Parámetros

targetGroup

ID único que identifica el grupo al que añadir al jugador. Los ID de grupos están definidos por el desarrollador.

Tipo: entero

Obligatorio: sí

Valor devuelto

Este método no devuelve nada. Dado que esta solicitud se envía mediante el método de entrega de confianza (TCP), las solicitudes erróneas activan la devolución de llamada OnError().

LeaveGroup()

Elimina al jugador de la pertenencia a un grupo especificado. Cuando ya no esté en el grupo, el jugador no recibe los mensajes enviados al grupo y no puede enviar mensajes a todo el grupo.

Sintaxis

```
public void LeaveGroup(int targetGroup)
```

Parámetros

targetGroup

ID único que identifica el grupo del que eliminar al jugador. Los ID de grupos están definidos por el desarrollador.

Tipo: entero

Obligatorio: sí

Valor devuelto

Este método no devuelve nada. Dado que esta solicitud se envía mediante el método de entrega de confianza (TCP), las solicitudes erróneas activan la devolución de llamada OnError().

RequestGroupMembership()

Solicita que se envíe una lista de los jugadores en el grupo especificado al cliente del juego. Cualquier jugador puede solicitar esta información, independientemente de si son miembros del grupo o no. En respuesta a esta solicitud, la lista de miembros se envía al cliente a través de una devolución de llamada OnGroupMembershipUpdated().

Sintaxis

```
public void RequestGroupMembership(int targetGroup)
```

Parámetros

targetGroup

ID único que identifica el grupo para el que obtener información de pertenencia. Los ID de grupos están definidos por el desarrollador.

Tipo: entero

Obligatorio: sí

Valor devuelto

Este método no devuelve nada.

Referencia de la API de cliente de Realtime Servers (C#) de : Devoluciones de llamadas asíncronas

Utilice esta referencia de la API de cliente de Realtime de C# para que le ayude a preparar el juego multijugador para utilizarlo con servidores de Realtime Servers implementados en flotas de Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Preparación del servidor de Realtime.

- Acciones síncronas
- Devoluciones de llamadas asíncronas
- Tipos de datos

Un cliente de juego necesita implementar estos métodos de devolución de llamadas para responder a eventos. El servidor de Realtime invoca estas devoluciones de llamadas para enviar información relacionada con juego al cliente del juego. Las devoluciones de llamada para los mismos eventos también se pueden implementar con lógica de juego personalizada en el script del servidor de Realtime. Consulte Devoluciones de llamadas de script para Realtime Servers.

los métodos de devolución de llamada se definen en `ClientEvents.cs`.

OnOpen()

Se invoca cuando el proceso del servidor acepta la solicitud de conexión del cliente de juego y abre una conexión.

Sintaxis

```
public void OnOpen()
```

Parámetros

Este método no toma parámetros.

Valor devuelto

Este método no devuelve nada.

OnClose()

Se invoca cuando el proceso de servidor termina la conexión con el cliente del juego, como después de que termina una sesión de juego.

Sintaxis

```
public void OnClose()
```

Parámetros

Este método no toma parámetros.

Valor devuelto

Este método no devuelve nada.

OnError()

Se invoca cuando se produce un error para una solicitud de la API del cliente de Realtime. Esta devolución de llamada se puede personalizar para administrar una variedad de errores de conexión.

Sintaxis

```
private void OnError(byte[] args)
```

Parámetros

Este método no toma parámetros.

Valor devuelto

Este método no devuelve nada.

OnDataReceived()

Se invoca cuando el cliente del juego recibe un mensaje desde el servidor de Realtime. Este es el método principal por el un cliente de juego recibe mensajes y notificaciones.

Sintaxis

```
public void OnDataReceived(DataReceivedEventArgs dataReceivedEventArgs)
```

Parámetros

dataReceivedEventArgs

Información relacionada con la actividad de mensajes.

Escriba: DataReceivedEventArgs

Obligatorio: sí

Valor devuelto

Este método no devuelve nada.

OnGroupMembershipUpdated()

Se invoca cuando se ha actualizado la pertenencia a un grupo al que pertenece el jugador. Esta devolución de llamada también se invoca cuando un cliente llama a `RequestGroupMembership`.

Sintaxis

```
public void OnGroupMembershipUpdated(GroupMembershipEventArgs groupMembershipEventArgs)
```

Parámetros

groupMembershipEventArgs

Información relacionada con la actividad de pertenencia a un grupo.

Escriba: GroupMembershipEventArgs

Obligatorio: sí

Valor devuelto

Este método no devuelve nada.

Referencia de la API de cliente de Realtime Servers (C#): Tipos de datos

Esta referencia de la API de cliente de Realtime de C# puede ayudarle a preparar el juego multijugador para utilizarlo con servidores de Realtime Servers implementados en flotas de Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Preparación del servidor de Realtime.

- Acciones síncronas
- Devoluciones de llamadas asíncronas
- Tipos de datos

ClientConfiguration

Información sobre cómo se conecta el cliente de juegos a un servidor de Realtime.

Contenido

ConnectionType

Tipo de conexión cliente/servidor que se va a utilizar, ya sea segura o no segura. Si no especifica un tipo de conexión, el valor predeterminado es conexión no segura.

> **Note**
>
> Al conectarse a un servidor Realtime en una flota protegida con un certificado TLS, debe utilizar el valor RT_OVER_WSS_DTLS_TLS12.

Tipo: un valor de enum de `ConnectionType`.

Obligatorio: no

ConnectionToken

Información sobre el cliente de juegos o jugador que solicita una conexión con un servidor de Realtime.

Contenido

playerSessionId

ID único emitido por Amazon GameLift cuando se crea una nueva sesión de jugador. El ID de sesión de un jugador se especifica en un objeto `PlayerSession`, que se devuelve como respuesta a una llamada de cliente a las acciones de la API de GameLift StartGameSessionPlacement, CreateGameSession, DescribeGameSessionPlacement o DescribePlayerSessions.

Tipo: String

Obligatorio: sí

payload

Información definida por el desarrollador que se comunica al servidor de Realtime durante la conexión. Esto incluye cualquier dato arbitrario que se pudiera usar para un mecanismo de inicio de sesión personalizado. Por ejemplo, una carga podría facilitar información de autenticación para que la procese el script del servidor de Realtime antes de permitir conectarse a un cliente.

Tipo: matriz de bytes

Obligatorio: no

RTMessage

Contenido e información de entrega para un mensaje. Un mensaje debe especificar un jugador o grupo destinatario.

Contenido

opCode

Código de operación definido por el desarrollador que identifica un evento o acción del juego, como un movimiento del jugador o una notificación del servidor. El opCode de un mensaje ofrece contexto para la carga de datos que se facilita. Los mensajes que se crean con NewMessage() ya tienen establecido el código de operación, pero se pueden modificar en cualquier momento.

Tipo: entero

Obligatorio: sí

targetPlayer

ID único que identifica al jugador que es el destinatario previsto del mensaje que se envía. El destinatario puede ser el propio servidor (con el ID de servidor) u otro jugador (con el ID del jugador).

Tipo: entero

Obligatorio: no

targetGroup

ID único que identifica al grupo que es el destinatario previsto del mensaje que se envía. Los ID de grupos están definidos por el desarrollador.

Tipo: entero

Obligatorio: no

deliveryIntent

Indica si se va a enviar el mensaje utilizando la conexión TCP de confianza o utilizando el canal UDP rápido. Mensajes creados con NewMessage().

Tipo: DeliveryIntent enum

Valores válidos: FAST | RELIABLE

Obligatorio: sí

payload

Contenido del mensaje. Esta información se estructura según sea necesario para que la procese el cliente del juego en función del código de operación adjunto. Podría contener datos sobre el estado del juego u otra información que se deba comunicar entre los clientes del juego o entre un cliente del juego y el servidor Realtime.

Tipo: matriz de bytes

Obligatorio: no

DataReceivedEventArgs

Datos facilitados con una devolución de llamada OnDataReceived().

Contenido

remitente

ID único que identifica la entidad (ID de jugador o ID de servidor) que originó el mensaje.

Tipo: entero

Obligatorio: sí

opCode

Código de operación definido por el desarrollador que identifica un evento o acción del juego, como un movimiento del jugador o una notificación del servidor. El opCode de un mensaje ofrece contexto para la carga de datos que se facilita.

Tipo: entero

Obligatorio: sí

data

Contenido del mensaje. Esta información se estructura según sea necesario para que la procese el cliente del juego en función del código de operación adjunto. Podría contener datos sobre el estado del juego u otra información que se deba comunicar entre los clientes del juego o entre un cliente del juego y el servidor Realtime.

Tipo: matriz de bytes

Obligatorio: no

GroupMembershipEventArgs

Datos facilitados con una devolución de llamada OnGroupMembershipUpdated().

Contenido

remitente

ID único que identifica al jugador que solicitó una actualización de su pertenencia al grupo.

Tipo: entero

Obligatorio: sí

opCode

Código de operación definido por el desarrollador que identifica un evento o acción del juego.

Tipo: entero

Obligatorio: sí

groupId

ID único que identifica al grupo que es el destinatario previsto del mensaje que se envía. Los ID de grupos están definidos por el desarrollador.

Tipo: entero

Obligatorio: sí

playerId

Lista de ID de jugadores que sean miembros del grupo especificado.

Tipo: matriz de enteros

Obligatorio: sí

Enums

Los enums definidos para el SDK de cliente de Realtime se definen así:

ConnectionStatus

- CONECTADO: el cliente de juegos está conectado al servidor de Realtime únicamente mediante una conexión TCP. Todos los mensajes, independientemente del intento de entrega, se envían por TCP.
- CONNECTED_SEND_FAST: el cliente de juegos está conectado al servidor de Realtime mediante una conexión TCP y UDP. Sin embargo, la capacidad de recibir mensajes a través de UDP aún no está verificada; por lo tanto, todos los mensajes enviados al cliente del juego utilizan TCP.
- CONNECTED_SEND_AND_RECEIVE_FAST: el cliente de juegos está conectado al servidor de Realtime mediante una conexión TCP y UDP. El cliente del juego puede enviar y recibir mensajes mediante TCP o UDP.
- CONECTANDO: el cliente del juego ha enviado una solicitud de conexión y el servidor de Realtime la está procesando.

- DISCONNECTED_CLIENT_CALL: el cliente del juego se desconectó del servidor de Realtime en respuesta a una solicitud de Disconnect() del cliente del juego.
- DESCONECTADO: el cliente de juegos se desconectó del servidor de Realtime por un motivo distinto a una llamada de desconexión del cliente.

ConnectionType

- RT_OVER_WSS_DTLS_TLS12: tipo de conexión segura.

  Para su uso con servidores Realtime que se ejecutan en una flota de GameLift con un certificado TLS generado. Cuando se utiliza una conexión segura, el tráfico TCP se cifra con TLS 1.2 y el tráfico UDP se cifra con DTLS 1.2.
- RT_OVER_WS_UDP_UNSECURED: tipo de conexión no segura.
- RT_OVER_WEBSOCKET: tipo de conexión no segura. Este valor ya no es el valor preferido.

DeliveryIntent

- RÁPIDO: se entrega mediante un canal UDP.
- FIABLE: se entrega mediante una conexión TCP.

Referencia de scripts de Servidores en tiempo real de Amazon GameLift

Utilice estos recursos para crear lógica personalizada en sus scripts de Realtime.

Temas

- Devoluciones de llamadas de script para Realtime Servers
- Interfaz de Realtime Servers

Devoluciones de llamadas de script para Realtime Servers

Puede proporcionar lógica personalizada para responder a eventos mediante la implementación de estas devoluciones de llamada en su script de Realtime.

Init

Inicializa el servidor de Realtime y recibe una interfaz de servidor en tiempo real.

Sintaxis

```
init(rtsession)
```

onMessage

Se invoca cuando un mensaje recibido se envía al servidor.

Sintaxis

```
onMessage(gameMessage)
```

onHealthCheck

Se invoca para establecer el estado de la sesión de juego. De forma predeterminada, el estado se encuentra en buen estado (o `true`). Esta devolución de llamada se pueden implementar para realizar comprobaciones de estado personalizadas y devolver un estado.

Sintaxis

```
onHealthCheck()
```

onStartGameSession

Se invoca cuando comienza una nueva sesión de juego, con un objeto de sesión de juego transferido.

Sintaxis

```
onStartGameSession(session)
```

onProcessTerminate

Se invoca cuando el servicio de Amazon GameLift finaliza el proceso del servidor. Esto puede actuar como un disparador para salir de forma correcta de la sesión de juego. No es necesario llamar a `processEnding().`

Sintaxis

```
onProcessTerminate()
```

onPlayerConnect

Se invoca cuando un jugador solicita una conexión y ha pasado una validación inicial.

Sintaxis

```
onPlayerConnect(connectMessage)
```

onPlayerAccepted

Se invoca cuando se acepta una conexión de jugador.

Sintaxis

```
onPlayerAccepted(player)
```

onPlayerDisconnect

Se invoca cuando un jugador se desconecta de la sesión del juego, ya sea enviando una solicitud de desconexión o por otros métodos.

Sintaxis

```
onPlayerDisconnect(peerId)
```

onProcessStarted

Se invoca cuando se inicia un proceso de servidor. Esta devolución de llamada permite que el script realice las tareas personalizadas necesarias para la preparación para alojar una sesión de juego.

Sintaxis

```
onProcessStarted(args)
```

onSendToPlayer

Se invoca cuando se recibe un mensaje en el servidor de un jugador para entregarlo a otro jugador. Este proceso se ejecuta antes de que se entregue el mensaje.

Sintaxis

```
onSendToPlayer(gameMessage)
```

onSendToGroup

Se invoca cuando se recibe un mensaje en el servidor de un jugador para entregarlo a un grupo. Este proceso se ejecuta antes de que se entregue el mensaje.

Sintaxis

```
onSendToGroup(gameMessage))
```

onPlayerJoinGroup

Se invoca cuando un jugador envía una solicitud para unirse a un grupo.

Sintaxis

```
onPlayerJoinGroup(groupId, peerId)
```

onPlayerLeaveGroup

Se invoca cuando un jugador envía una solicitud para abandonar un grupo.

Sintaxis

```
onPlayerLeaveGroup(groupId, peerId)
```

Interfaz de Realtime Servers

Cuando un script de Realtime se inicializa, se devuelve una interfaz para el servidor de Realtime. En este tema se describen las propiedades y los métodos disponibles a través de la interfaz. Obtenga más información sobre cómo escribir scripts de Realtime y vea un ejemplo de script detallado en Creación de un script de Realtime.

La interfaz de Realtime proporciona acceso a los siguientes objetos:

- session
- player
- gameMessage
- configuration

Objeto de sesión de Realtime

Utilice estos métodos para acceder a información relativa al servidor y llevar a cabo acciones relacionadas con los servidores.

getPlayers()

Recupera una lista de ID de homólogos para los jugadores que se encuentran conectados actualmente a la sesión de juego. Devuelve una matriz de objetos de jugador.

Sintaxis

```
rtSession.getPlayers()
```

broadcastGroupMembershipUpdate()

Activa la entrega de una lista de miembros de grupo actualizada al grupo de jugadores. Especifica los miembros a los que transmitir (groupIdToBroadcast) y el grupo que recibe la actualización (targetGroupId). Los ID de grupo deben ser un número entero positivo o «-1» para indicar todos los grupos. Consulte Ejemplo del script de Realtime Servers para ver un ejemplo de ID de grupo definidos por el usuario.

Sintaxis

```
rtSession.broadcastGroupMembershipUpdate(groupIdToBroadcast, targetGroupId)
```

getServerId()

Recupera el identificador de ID de homólogo único del servidor, que se utiliza para dirigir mensajes al servidor.

Sintaxis

```
rtSession.getServerId()
```

getAllPlayersGroupId()

Recupera el ID del grupo para el grupo predeterminado que contiene todos los jugadores conectados actualmente a la sesión de juego.

Sintaxis

```
rtSession.getAllPlayersGroupId()
```

processEnding()

Activa el servidor de Realtime para terminar el servidor de juegos. A esta función se debe llamar desde el script de Realtime para salir de forma correcta de una sesión de juego.

Sintaxis

```
rtSession.processEnding()
```

getGameSessionId()

Recupera el ID exclusivo de la sesión de juego que se está ejecutando actualmente.

Sintaxis

```
rtSession.getGameSessionId()
```

getLogger()

Recupera la interfaz para el registro. Utilice esta opción para registrar declaraciones que se capturarán en los registros de la sesión de juego. El registrador admite el uso de declaraciones "info", "warn" y "error". Por ejemplo: `logger.info("<string>")`.

Sintaxis

```
rtSession.getLogger()
```

sendMessage()

Envía un mensaje, creado con `newTextGameMessage` o `newBinaryGameMessage`, desde el servidor de Realtime a un jugador destinatario con el canal UDP. Identifique el destinatario utilizando el ID de homólogo del jugador.

Sintaxis

```
rtSession.sendMessage(gameMessage, targetPlayer)
```

sendGroupMessage()

Envía un mensaje, creado con `newTextGameMessage` o `newBinaryGameMessage`, desde el servidor de Realtime a todos los jugadores en un grupo de jugadores mediante el canal UDP. Los ID de grupo deben ser un número entero positivo o «-1» para indicar todos los grupos. Consulte Ejemplo del script de Realtime Servers para ver un ejemplo de ID de grupo definidos por el usuario.

Sintaxis

```
rtSession.sendGroupMessage(gameMessage, targetGroup)
```

sendReliableMessage()

Envía un mensaje, creado con `newTextGameMessage` o `newBinaryGameMessage`, desde el servidor de Realtime a un jugador destinatario con el canal TCP. Identifique el destinatario utilizando el ID de homólogo del jugador.

Sintaxis

```
rtSession.sendReliableMessage(gameMessage, targetPlayer)
```

sendReliableGroupMessage()

Envía un mensaje, creado con `newTextGameMessage` o `newBinaryGameMessage`, desde el servidor de Realtime a todos los jugadores de un grupo de jugadores mediante el canal TCP. Los ID de grupo deben ser un número entero positivo o «-1» para indicar todos los grupos. Consulte Ejemplo del script de Realtime Servers para ver un ejemplo de ID de grupo definidos por el usuario.

Sintaxis

```
rtSession.sendReliableGroupMessage(gameMessage, targetGroup)
```

newTextGameMessage()

Crea un nuevo mensaje que contiene texto, que se va a enviar desde el servidor a los destinatarios jugadores mediante las funciones SendMessage. El formato del mensaje es similar al formato utilizado en el SDK de cliente de Realtime (consulte RTMessage). Devuelve un objeto `gameMessage`.

Sintaxis

```
rtSession.newTextGameMessage(opcode, sender, payload)
```

newBinaryGameMessage()

Crea un nuevo mensaje que contiene datos binarios, que se va a enviar desde el servidor a los destinatarios jugadores mediante las funciones SendMessage. El formato del mensaje es similar al formato utilizado en el SDK de cliente de Realtime (consulte RTMessage). Devuelve un objeto `gameMessage`.

Sintaxis

```
rtSession.newBinaryGameMessage(opcode, sender, binaryPayload)
```

Objeto del jugador

Información de acceso relacionada con el jugador.

player.peerId

El ID único que se asigna a un cliente de juegos cuando se conecta al servidor de Realtime y se une a la sesión de juego.

player.playerSessionId

El ID de sesión de jugador al que ha hecho referencia el cliente de juegos al conectarse al servidor de Realtime y unirse a la sesión de juego.

Objeto de mensaje de juego

Utilice estos métodos para acceder a los mensajes que recibe el servidor de Realtime. Los mensajes recibidos desde los clientes de juego tienen la estructura RTMessage.

getPayloadAsText()

Obtiene la carga del mensaje del juego en forma de texto.

Sintaxis

```
gameMessage.getPayloadAsText()
```

gameMessage.opcode

Código de operación incluido en un mensaje.

gameMessage.payload

Carga contenida en un mensaje. Puede ser texto o binario.

gameMessage.sender

ID de homólogo del cliente de juego que envió un mensaje.

gameMessage.reliable

Valor booleano que indica si el mensaje se envió a través de TCP (true) o UDP (false).

Objeto de configuración

El objeto de configuración se puede utilizar para anular las configuraciones predeterminadas.

configuration.maxPlayers

El número máximo de conexiones de cliente/servidor que Realtime Servers puede aceptar.

El valor predeterminado es 32.

configuration.pingIntervalTime

Intervalo de tiempo en milisegundos en el que el servidor intentará enviar un ping a todos los clientes conectados para verificar que las conexiones están en buen estado.

El valor predeterminado es 300 ms.

Referencia del SDK del servidor de Amazon GameLift

Esta sección contiene la documentación de referencia del SDK del servidor de Amazon GameLift. Utilice el SDK del servidor para integrar los servidores de juegos personalizados para comunicarse con el servicio de Amazon GameLift.

Temas

- Referencia del SDK del servidor de Amazon GameLift para C++

- Referencia del SDK del servidor de Amazon GameLift para C#
- Referencia del SDK del servidor de Amazon GameLift para Go
- Referencia del SDK del servidor de Amazon GameLift para Unreal Engine

Referencia del SDK del servidor de Amazon GameLift para C++

Puede utilizar esta referencia del SDK del servidor C++ de Amazon GameLift como ayuda para preparar su juego multijugador para su uso con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

Temas

- Referencia del SDK del servidor Amazon GameLift 5.x para C++
- Referencia del SDK del servidor C++ de Amazon GameLift 3.x

Referencia del SDK del servidor Amazon GameLift 5.x para C++

Esta referencia del SDK del servidor C++ de Amazon GameLift 5.x como ayuda para preparar su juego multijugador para su uso con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

> **Note**
>
> En este tema se describe la API de C++ de Amazon GameLift que puede utilizar al realizar la compilación con la biblioteca estándar de C++ (`std`). En concreto, esta documentación se aplica al código que se compila con la opción `-DGAMELIFT_USE_STD=1`.

Temas

- Referencia 5.x GameLift del SDK del servidor Amazon (C++): acciones
- Referencia GameLift del SDK de Amazon Server (C++): tipos de datos

Referencia 5.x GameLift del SDK del servidor Amazon (C++): acciones

Puedes usar esta referencia del SDK del servidor GameLift C++ de Amazon como ayuda para preparar tu juego multijugador para usarlo con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

> **Note**
>
> En este tema se describe la API de Amazon GameLift C++ que puede utilizar al compilar con la biblioteca estándar de C++ (`std`). En concreto, esta documentación se aplica al código que se compila con la opción `-DGAMELIFT_USE_STD=1`.

Acciones

- GetSdkVersion()
- InitSDK()
- InitSDK()
- ProcessReady()
- ProcessReadyAsync()
- ProcessEnding()
- ActivateGameSession()
- UpdatePlayerSessionCreationPolicy()
- GetGameSessionId()
- GetTerminationTime()
- AcceptPlayerSession()
- RemovePlayerSession()
- DescribePlayerSessions()
- StartMatchBackfill()
- StopMatchBackfill()
- GetComputeCertificate()
- GetFleetRoleCredentials()
- Destroy()

GetSdkVersion()

Devuelve el número de versión actual del SDK integrado en el proceso del servidor.

Sintaxis

```
Aws::GameLift::AwsStringOutcome Server::GetSdkVersion();
```

Valor devuelto

Si funciona correctamente, devuelve la versión del SDK actual como objeto the section called "AwsStringOutcome". El objeto devuelto incluye el número de versión (por ejemplo, `5.0.0`). Si no funciona, devuelve un mensaje de error.

Ejemplo

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =
 Aws::GameLift::Server::GetSdkVersion();
```

InitSDK()

Inicializa el Amazon GameLift SDK para una flota de EC2 gestionada. Llama a este método en el momento del lanzamiento, antes de que GameLift se produzca cualquier otra inicialización relacionada con Amazon. Este método lee los parámetros del servidor del entorno anfitrión para configurar la comunicación entre el servidor y el GameLift servicio de Amazon.

Sintaxis

```
Server::InitSDKOutcome Server::initSdkOutcome = InitSDK();
```

Valor devuelto

Devuelve un objeto the section called "InitSDKOutcome" que indica si el proceso del servidor está listo para llamar a ProcessReady().

Ejemplo

```
//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()

Inicializa el Amazon GameLift SDK para una Anywhere flota. Llama a este método en el momento del lanzamiento, antes de que GameLift se produzca cualquier otra inicialización relacionada con Amazon. Este método requiere parámetros de servidor explícitos para configurar la comunicación entre el servidor y el GameLift servicio de Amazon.

Sintaxis

```
Server::InitSDKOutcome Server::initSdkOutcome = InitSDK(serverParameters);
```

Parámetros

ServerParameters

Para inicializar un servidor de juegos en una GameLift Anywhere flota de Amazon, construye un `ServerParameters` objeto con la siguiente información:

- La URL WebSocket utilizada para conectarte a tu servidor de juegos.
- El ID del proceso utilizado para alojar su servidor de juegos.
- El ID del proceso utilizado para alojar los procesos del servidor de juegos.
- El ID de la GameLift flota de Amazon que contiene tu ordenador de GameLift Anywhere Amazon.
- El token de autorización generado por la GameLift operación de Amazon.

Valor devuelto

Devuelve un objeto the section called “InitSDKOutcome” que indica si el proceso del servidor está listo para llamar a ProcessReady().

> **Note**
>
> Si las llamadas a `InitSDK()` no funcionan en las compilaciones de juegos implementadas en las flotas de Anywhere, compruebe el parámetro `ServerSdkVersion` que se utiliza al crear el recurso de compilación. Debe establecer este valor de forma explícita en la versión del SDK del servidor en uso. El valor predeterminado de este parámetro es 4.x, que no es compatible. Para resolver este problema, cree una compilación nueva e impleméntela en una flota nueva.

Ejemplo

GameLift AnywhereEjemplo de 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()

Notifica a Amazon de GameLift que el proceso del servidor está listo para albergar sesiones de juego. Llame a este método después de invocar InitSDK(). Se debe llamar a este método solo una vez por proceso.

Sintaxis

```
GenericOutcome ProcessReady(const Aws::GameLift::Server::ProcessParameters
&processParameters);
```

Parámetros

processParameters

Es un objeto ProcessParameters que comunica la siguiente información acerca del proceso del servidor:

- Nombres de los métodos de devolución de llamada implementados en el código del servidor del juego que el GameLift servicio de Amazon invoca para comunicarse con el proceso del servidor.
- Número de puerto de escucha del servidor de proceso.

- Ruta a cualquier archivo específico de la sesión de juego que quieras que Amazon capture y GameLift almacene.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo ilustra las implementaciones tanto de la función de llamada ProcessReady() como de la función de delegación.

```
// Set parameters and call ProcessReady
std::string serverLog("serverOut.log");        // Example of a log file written by the
 game server
std::vector<std::string> logPaths;
logPaths.push_back(serverLog);
int listenPort = 9339;

Aws::GameLift::Server::ProcessParameters processReadyParameter =
  Aws::GameLift::Server::ProcessParameters(
    std::bind(&Server::onStartGameSession, this, std::placeholders::_1),
    std::bind(&Server::onProcessTerminate, this),
    std::bind(&Server::OnHealthCheck, this),
    std::bind(&Server::OnUpdateGameSession, this),
    listenPort,
    Aws::GameLift::Server::LogParameters(logPaths)
  );

Aws::GameLift::GenericOutcome outcome =
  Aws::GameLift::Server::ProcessReady(processReadyParameter);

// Implement callback functions
void Server::onStartGameSession(Aws::GameLift::Model::GameSession myGameSession)
{
  // game-specific tasks when starting a new game session, such as loading map
  GenericOutcome outcome =
    Aws::GameLift::Server::ActivateGameSession (maxPlayers);
}

void Server::onProcessTerminate()
{
  // game-specific tasks required to gracefully shut down a game session,
```

```
  // such as notifying players, preserving game state data, and other cleanup
  GenericOutcome outcome = Aws::GameLift::Server::ProcessEnding();
}

bool Server::onHealthCheck()
{
  bool health;
  // complete health evaluation within 60 seconds and set health
  return health;
}
```

ProcessReadyAsync()

Notifica al GameLift servicio de Amazon que el proceso del servidor está listo para albergar sesiones de juego. Este método debe llamarse después de que el proceso del servidor esté listo para alojar una sesión de juego. Los parámetros especifican los nombres de las funciones de devolución de llamada GameLift a las que Amazon debe llamar en determinadas circunstancias. El código de servidor de juegos debe implementar estas funciones.

La llamada es asíncrona. Para realizar una llamada síncrona, utilice ProcessReady(). Consulte Inicialización del proceso del servidor para obtener más detalles.

Sintaxis

```
GenericOutcomeCallable ProcessReadyAsync(
    const Aws::GameLift::Server::ProcessParameters &processParameters);
```

Parámetros

processParameters

Es un objeto ProcessParameters que comunica la siguiente información acerca del proceso del servidor:

- Nombres de los métodos de devolución de llamada implementados en el código del servidor del juego que el GameLift servicio de Amazon invoca para comunicarse con el proceso del servidor.
- Número de puerto de escucha del servidor de proceso.
- Ruta a cualquier archivo específico de la sesión de juego que quieras que Amazon capture y GameLift almacene.

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
// Set parameters and call ProcessReady
std::string serverLog("serverOut.log");        // This is an example of a log file
 written by the game server
std::vector<std::string> logPaths;
logPaths.push_back(serverLog);
int listenPort = 9339;

Aws::GameLift::Server::ProcessParameters processReadyParameter =
  Aws::GameLift::Server::ProcessParameters(std::bind(&Server::onStartGameSession, this,
 std::placeholders::_1),
  std::bind(&Server::onProcessTerminate, this), std::bind(&Server::OnHealthCheck,
 this),
  std::bind(&Server::OnUpdateGameSession, this), listenPort,
 Aws::GameLift::Server::LogParameters(logPaths));

Aws::GameLift::GenericOutcomeCallable outcome =
  Aws::GameLift::Server::ProcessReadyAsync(processReadyParameter);

// Implement callback functions
void onStartGameSession(Aws::GameLift::Model::GameSession myGameSession)
{
  // game-specific tasks when starting a new game session, such as loading map
  GenericOutcome outcome = Aws::GameLift::Server::ActivateGameSession (maxPlayers);
}

void onProcessTerminate()
{
  // game-specific tasks required to gracefully shut down a game session,
  // such as notifying players, preserving game state data, and other cleanup
  GenericOutcome outcome = Aws::GameLift::Server::ProcessEnding();
}

bool onHealthCheck()
{
  // perform health evaluation and complete within 60 seconds
  return health;
}
```

ProcessEnding()

Notifica a Amazon de GameLift que el proceso del servidor está finalizando. Utiliza este método después de realizar todas las demás tareas de limpieza (incluido el cierre de la sesión de juego activa) y antes de finalizar el proceso. Según el resultado de `ProcessEnding()`, el proceso finaliza con éxito (0) o error (-1) y genera un evento de flota. Si el proceso termina con un error, se generará el evento de flota. `SERVER_PROCESS_TERMINATED_UNHEALTHY`

Sintaxis

```
Aws::GameLift::GenericOutcome processEndingOutcome =
 Aws::GameLift::Server::ProcessEnding();
```

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

En este ejemplo, se llama a `ProcessEnding()` y `Destroy()` antes de finalizar el proceso del servidor con un código de salida correcto o erróneo.

```
Aws::GameLift::GenericOutcome processEndingOutcome =
 Aws::GameLift::Server::ProcessEnding();
Aws::GameLift::Server::Destroy();

// Exit the process with success or failure
if (processEndingOutcome.IsSuccess()) {
  exit(0);
}
else {
  cout << "ProcessEnding() failed. Error: " <<
 processEndingOutcome.GetError().GetErrorMessage();
  exit(-1);
}
```

ActivateGameSession()

Notifica a Amazon GameLift que el proceso del servidor ha activado una sesión de juego y ya está listo para recibir las conexiones de los jugadores. Esta acción debe llamarse como parte de la

función de devolución de llamada `onStartGameSession()`, después de la inicialización de todas las sesiones de juego.

Sintaxis

```
Aws::GameLift::GenericOutcome activateGameSessionOutcome =
 Aws::GameLift::Server::ActivateGameSession();
```

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo muestra cómo se llama a `ActivateGameSession()` como parte de la función de delegación `onStartGameSession()`.

```
void onStartGameSession(Aws::GameLift::Model::GameSession myGameSession)
{
  // game-specific tasks when starting a new game session, such as loading map
  GenericOutcome outcome = Aws::GameLift::Server::ActivateGameSession();
}
```

UpdatePlayerSessionCreationPolicy()

Actualiza la capacidad de la sesión de juego actual para aceptar sesiones de jugador nuevas. Una sesión de juego se puede configurar para que acepte o deniegue todas las sesiones nuevas de los jugadores.

Sintaxis

```
GenericOutcome
 UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::PlayerSessionCreationPolicy
 newPlayerSessionPolicy);
```

Parámetros

playerCreationSessionPolítica

Tipo: valor de `PlayerSessionCreationPolicy` enumeración.

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo establece la política de participación en la sesión de juego actual para aceptar todos los jugadores.

```
Aws::GameLift::GenericOutcome outcome =

 Aws::GameLift::Server::UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::PlayerSessionCr
```

GetGameSessionId()

Recupera el ID de la sesión de juego alojada por el proceso del servidor.

En el caso de los procesos inactivos que no se activan con una sesión de juego, la llamada devuelve the section called “GameLiftError”.

Sintaxis

```
AwsStringOutcome GetGameSessionId()
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Si funciona correctamente, devuelve el ID de sesión del juego como objeto the section called “AwsStringOutcome”. Si no funciona, devuelve un mensaje de error.

En el caso de los procesos inactivos que no se activan con una sesión de juego, la llamada devuelve `Success=True` y `GameSessionId=""`.

Ejemplo

```
Aws::GameLift::AwsStringOutcome sessionIdOutcome =
 Aws::GameLift::Server::GetGameSessionId();
```

GetTerminationTime()

Devuelve la hora a la que está programada el cierre de un proceso de servidor, si hay una hora de terminación disponible. Un proceso de servidor toma medidas después de recibir una `onProcessTerminate()` llamada de Amazon GameLift. Amazon GameLift solicita `onProcessTerminate()` por los siguientes motivos:

- Cuando el proceso del servidor ha informado de un mal estado de salud o no ha respondido a Amazon GameLift.
- Al finalizar la instancia durante un evento de reducción vertical.
- Cuando se finaliza una instancia debido a la interrupción de una instancia de spot.

Sintaxis

```
AwsDateTimeOutcome GetTerminationTime()
```

Valor devuelto

Si el proceso se realiza correctamente, devuelve la hora de terminación como un objeto `AwsDateTimeOutcome`. El valor es la hora de terminación expresado en ciclos transcurridos desde `0001 00:00:00`. Por ejemplo, el valor de la fecha y hora `2020-09-13 12:26:40 -000Z` es igual a `637355968000000000` ciclos. Si no hay una hora de terminación disponible, devuelve un mensaje de error.

Si el proceso no ha recibido un ProcessParameters. OnProcessTerminate() devolución de llamada, se devuelve un mensaje de error. Para obtener más información sobre cómo cerrar un proceso de servidor, consulte Respuesta a una notificación de cierre del proceso del servidor.

Ejemplo

```
Aws::GameLift::AwsLongOutcome TermTimeOutcome =
 Aws::GameLift::Server::GetTerminationTime();
```

AcceptPlayerSession()

Notifica a Amazon GameLift que un jugador con el identificador de sesión de jugador especificado se ha conectado al proceso del servidor y necesita ser validado. Amazon GameLift verifica que

el identificador de sesión del jugador sea válido. Una vez validada la sesión del jugador, Amazon GameLift cambia el estado del espacio del jugador de RESERVADO a ACTIVO.

Sintaxis

```
GenericOutcome AcceptPlayerSession(String playerSessionId)
```

Parámetros

playerSessionId

ID único emitido por Amazon GameLift cuando se crea una nueva sesión de jugador.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

En este ejemplo, se administra una solicitud de conexión que incluye la validación y el rechazo de los ID de sesión de los jugadores no válidos.

```
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()

Notifica a Amazon GameLift que un jugador se ha desconectado del proceso del servidor. En respuesta, Amazon GameLift cambia el espacio del jugador para que esté disponible.

Sintaxis

```
GenericOutcome RemovePlayerSession(String playerSessionId)
```

Parámetros

`playerSessionId`

ID único emitido por Amazon GameLift cuando se crea una nueva sesión de jugador.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
Aws::GameLift::GenericOutcome disconnectOutcome =
 Aws::GameLift::Server::RemovePlayerSession(playerSessionId);
```

DescribePlayerSessions()

Recupera datos de sesión de jugador, incluida la configuración, los metadatos de la sesión y los datos de jugador. Utilice este método para obtener información sobre los siguientes elementos:

- Una sesión para un jugador
- Todas las sesiones del jugador en una sesión de juego
- Todas las sesiones de jugador están asociadas a un único ID de jugador

Sintaxis

```
DescribePlayerSessionsOutcome DescribePlayerSessions(DescribePlayerSessionsRequest
 describePlayerSessionsRequest)
```

Parámetros

DescribePlayerSessionsRequest

Un objeto the section called “DescribePlayerSessionsRequest” que describe las sesiones de jugador que recuperar.

Valor devuelto

Si funciona correctamente, devuelve un objeto the section called “DescribePlayerSessionsOutcome” que contiene un conjunto de objetos de sesión de jugador que se ajusta a los parámetros de la solicitud.

Ejemplo

En este ejemplo se solicitan todas las sesiones de jugador conectadas activamente a una sesión de juego específica. Al omitir NextTokeny establecer el valor límite en 10, Amazon GameLift devuelve los registros de las sesiones de los primeros 10 jugadores que coincidan con la solicitud.

```
// 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()

Envía una solicitud para encontrar nuevos jugadores para ranuras abiertas en una sesión de juego creada con FlexMatch. Para obtener más información, consulta la función FlexMatch de relleno.

Esta acción es asíncrona. Si se emparejan nuevos jugadores, Amazon GameLift proporciona datos actualizados de los emparejadores mediante la función de devolución de llamada. `OnUpdateGameSession()`

Un proceso del servidor solo puede tener una solicitud de reposición de emparejamiento activa a la vez. Para enviar una nueva solicitud, en primer lugar llame a StopMatchBackfill() para cancelar la solicitud original.

Sintaxis

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest
 startBackfillRequest);
```

Parámetros

StartMatchBackfillRequest

Un StartMatchBackfillRequest objeto que comunica la siguiente información:

- Un ID de ticket que se asignará a la solicitud de reposición. Esta información es opcional; si no se proporciona ningún identificador, Amazon GameLift generará uno.
- El creador de emparejamientos al que se enviará la solicitud. El ARN de configuración completo es obligatorio. Este valor se encuentra en los datos del emparejador de la sesión de juego.
- El ID de la sesión de juego que se va a reponer.
- Datos del emparejador disponibles para los jugadores actuales de la sesión de juego.

Valor devuelto

Devuelve un objeto the section called “StartMatchBackfillOutcome” con el ID del ticket de reposición de emparejamiento o un error con un mensaje de error.

Ejemplo

```
// Build a backfill request
std::vector<Player> players;
Aws::GameLift::Server::Model::StartMatchBackfillRequest startBackfillRequest;
startBackfillRequest.SetTicketId("1111aaaa-22bb-33cc-44dd-5555eeee66ff");  // optional,
 autogenerated if not provided
startBackfillRequest.SetMatchmakingConfigurationArn("arn:aws:gamelift:us-
west-2:111122223333:matchmakingconfiguration/MyMatchmakerConfig"); //from the game
 session matchmaker data
startBackfillRequest.SetGameSessionArn("the game session ARN");        // can use
 GetGameSessionId()
startBackfillRequest.SetPlayers(players);                                  // from the
 game session matchmaker data

// Send backfill request
Aws::GameLift::StartMatchBackfillOutcome backfillOutcome =
  Aws::GameLift::Server::StartMatchBackfill(startBackfillRequest);

// Implement callback function for backfill
void Server::OnUpdateGameSession(Aws::GameLift::Server::Model::GameSession gameSession,
 Aws::GameLift::Server::Model::UpdateReason updateReason, std::string backfillTicketId)
{
  // handle status messages
```

```
  // perform game-specific tasks to prep for newly matched players
}
```

StopMatchBackfill()

Cancela una solicitud de reposición de emparejamiento activa. Para obtener más información, consulta la función de FlexMatch relleno.

Sintaxis

```
GenericOutcome StopMatchBackfill (StopMatchBackfillRequest stopBackfillRequest);
```

Parámetros

StopMatchBackfillRequest

Un StopMatchBackfillRequest objeto que identifica el billete de emparejamiento que se va a cancelar:

- ID del ticket que se asignará a la solicitud de reposición
- El emparejador al que se envió la solicitud de reposición
- La sesión de juego asociada a la solicitud de reposición.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
// Set backfill stop request parameters

Aws::GameLift::Server::Model::StopMatchBackfillRequest stopBackfillRequest;
stopBackfillRequest.SetTicketId("1111aaaa-22bb-33cc-44dd-5555eeee66ff");
stopBackfillRequest.SetGameSessionArn("the game session ARN"); // can use
 GetGameSessionId()
stopBackfillRequest.SetMatchmakingConfigurationArn("arn:aws:gamelift:us-
west-2:111122223333:matchmakingconfiguration/MyMatchmakerConfig");
// from the game session matchmaker data

Aws::GameLift::GenericOutcome stopBackfillOutcome =
  Aws::GameLift::Server::StopMatchBackfill(stopBackfillRequest);
```

GetComputeCertificate()

Recupera la ruta al certificado TLS utilizado para cifrar la conexión de red entre el recurso GameLift Anywhere informático de Amazon y Amazon. GameLift Puedes usar la ruta del certificado al registrar tu dispositivo informático en una GameLift Anywhere flota de Amazon. Para obtener más información, consulte, RegisterCompute.

Sintaxis

```
GetComputeCertificateOutcome Server::GetComputeCertificate()
```

Valor devuelto

Devuelve the section called “GetComputeCertificateOutcome”.

Ejemplo

```
Aws::GameLift::GetComputeCertificateOutcome certificate =
 Aws::GameLift::Server::GetComputeCertificate();
```

GetFleetRoleCredentials()

Recupera las credenciales del rol de IAM que autorizan GameLift a Amazon a interactuar con otros. Servicios de AWS Para obtener más información, consulte Comunicación con otros recursos de AWS de sus flotas.

Sintaxis

```
GetFleetRoleCredentialsOutcome GetFleetRoleCredentials(GetFleetRoleCredentialsRequest
 request);
```

Parámetros

GetFleetRoleCredentialsRequest

Valor devuelto

Devuelve un objeto the section called “GetFleetRoleCredentialsOutcome”.

Ejemplo

```
// 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);
```

En este ejemplo, se muestra el uso del valor `RoleSessionName` opcional para asignar un nombre a la sesión de credenciales con fines de auditoría. Si no proporciona un nombre de sesión de rol, se utiliza el valor predeterminado «*`[fleet-id]`*-*`[host-id]`*».

```
// form the fleet credentials request
Aws::GameLift::Server::Model::GetFleetRoleCredentialsRequest
 getFleetRoleCredentialsRequest;
getFleetRoleCredentialsRequest.SetRoleArn("arn:aws:iam::123456789012:role/service-role/
exampleGameLiftAction");
getFleetRoleCredentialsRequest.SetRoleSessionName("MyFleetRoleSession");

Aws::GameLift::GetFleetRoleCredentialsOutcome credentials =
 Aws::GameLift::Server::GetFleetRoleCredentials(getFleetRoleCredentialsRequest);
```

Destroy()

Libera de memoria el SDK del servidor de GameLift juegos de Amazon. Como práctica recomendada, llame a este método después de `ProcessEnding()` y antes de finalizar el proceso. Si utilizas una flota de Anywhere y no vas a finalizar los procesos del servidor después de cada sesión de juego, llama `Destroy()` y, `InitSDK()` a continuación, reinicializa antes de notificar a Amazon GameLift que el proceso está listo para organizar una sesión de juego. `ProcessReady()`

Sintaxis

```
GenericOutcome Aws::GameLift::Server::Destroy();
```

Parámetros

No hay parámetros.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
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);
}
```

Referencia GameLift del SDK de Amazon Server (C++): tipos de datos

Puedes usar esta referencia del SDK del servidor GameLift C++ de Amazon como ayuda para preparar tu juego multijugador para usarlo con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

> **Note**
>
> En este tema se describe la API de Amazon GameLift C++ que puede utilizar al compilar con la biblioteca estándar de C++ (`std`). En concreto, esta documentación se aplica al código que se compila con la opción `-DGAMELIFT_USE_STD=1`.

Tipos de datos

- LogParameters
- ProcessParameters
- UpdateGameSession
- GameSession
- ServerParameters
- StartMatchBackfillRequest
- Jugador
- DescribePlayerSessionsRequest

- StopMatchBackfillRequest
- AttributeValue
- GetFleetRoleCredentialsRequest
- AwsLongOutcome
- AwsStringOutcome
- DescribePlayerSessionsOutcome
- DescribePlayerSessionsResult
- GenericOutcome
- GenericOutcomeCallable
- PlayerSession
- StartMatchBackfillOutcome
- StartMatchBackfillResult
- GetComputeCertificateOutcome
- GetComputeCertificateResult
- GetFleetRoleCredentialsOutcome
- GetFleetRoleCredentialsResult
- InitSDKOutcome
- GameLiftError
- Enums

LogParameters

Un objeto que identifica los archivos generados durante una sesión de juego que quieres que Amazon GameLift suba y almacene una vez finalizada la sesión de juego. El servidor del juego proporciona `LogParameters` a Amazon GameLift como parte de un `ProcessParameters` objeto en una ProcessReady() llamada.

Propiedades	Descripción
LogPaths	La lista de rutas de directorio a los archivos de registro del servidor de juegos que quieres que Amazon almacene GameLift para acceder a ellos en el futuro. El proceso del servidor

	genera esos archivos durante una sesión de juego. Defina los nombres y las rutas de los archivos en el servidor de juegos y almacénel os en el directorio raíz de compilación del juego. Las rutas del registro deben ser absolutas. Por ejemplo, si la compilación del juego almacena los registros de sesión de juego en una ruta del tipo `MyGame\sessionLogs\` , la ruta sería `c:\game\MyGame\sessionLogs` en una instancia de Windows. Tipo: `std:vector<std::string>` Obligatorio: no

ProcessParameters

Este tipo de datos contiene el conjunto de parámetros enviados a Amazon GameLift en unProcessReady().

Propiedades	Descripción
LogParameters	Un objeto con rutas de directorio a los archivos que se generan durante una sesión de juego. Amazon GameLift copia y almacena los archivos para acceder a ellos en el futuro. Tipo: `Aws::GameLift::Server:: LogParameters` Obligatorio: no
OnHealthCheck	La función de devolución de llamada que Amazon GameLift invoca para solicitar un informe de estado de salud al proceso del servidor. Amazon GameLift llama a esta

	función cada 60 segundos y espera 60 segundos para recibir una respuesta. El proceso del servidor devuelve `TRUE` si está en buen estado y `FALSE` si no. Si no se devuelve ninguna respuesta, Amazon GameLift registra el proceso del servidor como incorrecto. Tipo: `std::function<bool()> onHealthCheck` Obligatorio: no
OnProcessTerminate	La función de devolución de llamada que Amazon GameLift invoca para forzar el cierre del proceso del servidor. Tras llamar a esta función, Amazon GameLift espera 5 minutos a que se cierre el proceso del servidor y responde con una ProcessEnding() llamada antes de cerrar el proceso del servidor. Tipo: `std::function<void()> onProcessTerminate` Obligatorio: sí
OnRefreshConnection	El nombre de la función de devolución de llamada que Amazon GameLift invoca para actualizar la conexión con el servidor del juego. Tipo: `void OnRefreshConnectionDelegate()` Obligatorio: sí

OnStartGameSession	La función de devolución de llamada que Amazon GameLift invoca para activar una nueva sesión de juego. Amazon GameLift llama a esta función en respuesta a una solicitud de un cliente CreateGameSession. La función callback pasa un GameSessionobjeto, tal y como se define en la referencia de la GameLift API de Amazon. Tipo: `const std::function<void (Aws::GameLift::Model::GameSession)> onStartGameSession` Obligatorio: sí
OnUpdateGameSession	La función de devolución de llamada que Amazon GameLift invoca para pasar un objeto de sesión de juego actualizado al proceso del servidor. Amazon GameLift llama a esta función cuando se ha procesado una solicitud de relleno de partidas para proporcionar datos actualizados de los emparejadores. Transmite un GameSessionobjeto, una actualización de estado (`updateReason`) y el identificador del ticket de relleno del partido. Tipo: `std::function<void(Aws::GameLift::Server::Model::UpdateGameSession)> onUpdateGameSession` Obligatorio: no

Puerto	El número de puerto en el que escucha el proceso del servidor para recibir conexiones de jugador nuevas. El valor debe estar en el rango de puertos configurado para cualquier flota que implemente esa compilación de servidor de juegos. Este número de puerto se incluye en los objetos de sesión de juego y de jugador, que las sesiones de juego utilizan a la hora de conectarse a un proceso del servidor. Tipo: `Integer` Obligatorio: sí

UpdateGameSession

Este tipo de datos se actualiza a un objeto de sesión de juego, que incluye el motivo por el que se actualizó la sesión de juego y el ID del ticket de reposición correspondiente si la reposición se utiliza para reponer las sesiones de los jugadores en la sesión de juego.

Propiedades	Descripción
GameSession	Un GameSessionobjeto definido por la GameLift API de Amazon. El objeto `GameSession` contiene propiedades que describen una sesión de juego. Tipo: `Aws::GameLift::Server::GameSession` Obligatorio: sí
UpdateReason	El motivo por el que se actualiza la sesión de juego. Tipo: `Aws::GameLift::Server::UpdateReason`

Propiedades	Descripción
	Obligatorio: sí
BackfillTicketId	El ID de ticket de reposición que intenta actualizar la sesión de juego. Tipo: `std::string` Obligatorio: no

GameSession

Este tipo de datos proporciona detalles de una sesión de juego.

Propiedades	Descripción
GameSessionId	Un identificador único de la sesión de juego. El ARN de una sesión de juego tiene el siguiente formato: `arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token>` . Tipo: `std::string` Obligatorio: no
Nombre	Una etiqueta descriptiva de la sesión de juego. Tipo: `std::string` Obligatorio: no
FleetId	Un identificador único para la flota en la que se ejecuta la sesión de juego. Tipo: `std::string` Obligatorio: no

Propiedades	Descripción
MaximumPlayerSessionCount	El número máximo de conexiones de jugadores a la sesión de juego. Tipo: `int` Obligatorio: no
Puerto	El número de puerto de la sesión de juego. Para conectarse a un servidor de GameLift juegos de Amazon, una aplicación necesita tanto la dirección IP como el número de puerto. Tipo: `in` Obligatorio: no
IpAddress	La dirección IP de la sesión de juego. Para conectarse a un servidor de GameLift juegos de Amazon, una aplicación necesita tanto la dirección IP como el número de puerto. Tipo: `std::string` Obligatorio: no
GameSessionData	Un conjunto de propiedades de sesión de juego personalizadas, formateadas como un valor de una sola cadena. Tipo: `std::string` Obligatorio: no

Propiedades	Descripción
MatchmakerData	Información sobre el proceso de emparejamiento que se utilizó para crear la sesión de juego, en sintaxis JSON, con formato como cadena. Además de la configuración de emparejamiento utilizada, contiene datos sobre todos los jugadores asignados al emparejamiento, incluidos los atributos de los jugadores y las asignaciones de los equipos. Tipo: `std::string` Obligatorio: no
GameProperties	Un conjunto de propiedades personalizadas de una sesión de juego, con formato como pares clave-valor. Estas propiedades se pasan a una solicitud de iniciar una nueva sesión de juego. Tipo: `std :: vector < GameProperty >` Obligatorio: no

Propiedades	Descripción
DnsName	El identificador de DNS asignado a la instancia que ejecuta la sesión de juego. Los valores tienen formato siguiente: • Flotas habilitadas para TLS: `<unique identifier>.<region identifier>.amazongamelift.com` . • Flotas no habilitadas para TLS: `ec2-<unique identifier>.compute.amazonaws.com` . Cuando se conecte a una sesión de juego que se ejecute en una flota habilitada de TLS, debe utilizar el nombre de DNS, no la dirección IP. Tipo: `std::string` Obligatorio: no

ServerParameters

Información utilizada para mantener la conexión entre el servidor de juegos de una GameLift Anywhere flota de Amazon y el GameLift servicio de Amazon. Esta información se utiliza al inicializar nuevos procesos de servidor con InitSDK(). Para los servidores alojados en instancias EC2 GameLift gestionadas por Amazon, utilice un objeto vacío.

Propiedades	Descripción
webSocketUrl	`GameLiftServerSdkEndpoint` Amazon GameLift regresa cuando `RegisterCompute` buscas un recurso GameLift Anywhere informático de Amazon. Tipo: `std::string`

Propiedades	Descripción
	Obligatorio: sí
processId	Un identificador único registrado en el proceso de servidor que aloja el juego. Tipo: `std::string` Obligatorio: sí
hostId	El `HostID` es el `ComputeName` utilizado cuando registró el recurso informático. Para obtener más información, consulte, RegisterCompute. Tipo: `std::string` Obligatorio: sí
fleetId	El identificador único de la flota en la que está registrado el recurso informático. Para obtener más información, consulte, RegisterCompute. Tipo: `std::string` Obligatorio: sí
authToken	El token de autenticación generado por Amazon GameLift que autentica tu servidor en Amazon GameLift. Para obtener más información, consulte, GetComputeAuthToken. Tipo: `std::string` Obligatorio: sí

StartMatchBackfillRequest

Información utilizada para crear una solicitud de reposición de emparejamiento. El servidor del juego comunica esta información a Amazon GameLift en una StartMatchBackfill() llamada.

Propiedades	Descripción
GameSessionArn	Un identificador único de la sesión de juegos. El `GetGameSessionId` de la operación de la API devuelve el identificador en formato de ARN. Tipo: `std::string` Obligatorio: sí
MatchmakingConfigurationArn	Un identificador único, con formato de ARN, que el emparejador utiliza para esta solicitud . El ARN del emparejador para la sesión de juego original se encuentra en el objeto de sesión de juego en la propiedad de datos del emparejador. Puede obtener más información sobre los datos del emparejador en Trabajo con datos del emparejador. Tipo: `std::string` Obligatorio: sí
Players	Un conjunto de datos que representa a todos los jugadores que están en la sesión de juego. El creador de emparejamientos utiliza esta información para buscar nuevos jugadores que son idóneos para los jugadores actuales. Tipo: `std::vector<Player>` Obligatorio: sí

Propiedades	Descripción
TicketId	Un identificador único para un ticket de solicitud de emparejamiento o reposición de emparejamiento. Si no proporcionas un valor, Amazon GameLift generará uno. Use este identificador para realizar un seguimiento del estado del ticket de reposición de emparejamiento o cancelar la solicitud si es necesario. Tipo: `std::string` Obligatorio: no

Jugador

Este tipo de datos representa a un jugador en el sistema de emparejamiento. Al iniciar una solicitud de emparejamiento, un jugador tiene un ID de jugador, atributos y, posiblemente, datos de latencia. Amazon GameLift añade la información del equipo después de que se haya disputado un partido.

Propiedades	Descripción
LatencyInMS	Un conjunto de valores expresados en milisegundos que indican la cantidad de latencia que experimenta un jugador cuando se conecta a una ubicación. Si se utiliza esta propiedad, el jugador solo se empareja con las ubicaciones que aparecen. Si un creador de emparejamientos tiene una regla que evalúa la latencia de los jugadores, estos deben informar de la latencia para ser emparejados. Tipo: `Dictionary<string,int>` Obligatorio: no

Propiedades	Descripción
PlayerAttributes	Una colección de pares de clave-valor que contienen información del jugador para su uso en el emparejamiento. Las claves de atributos del jugador deben coincidir con las PlayerAtt ributes utilizadas en un conjunto de reglas de emparejamiento. Para obtener más información sobre los atributos de los jugadores, consulte Attribute Value. Tipo: `std::map<std::string,Attrib uteValue>` Obligatorio: no
PlayerId	Un identificador único de un jugador. Tipo: `std::string` Obligatorio: no
Equipo	El nombre del equipo al que está asignado el jugador en un emparejamiento. Defina el nombre del equipo se define en el conjunto de reglas de emparejamiento. Tipo: `std::string` Obligatorio: no

DescribePlayerSessionsRequest

Un objeto que especifica las sesiones de jugador que recuperar. El proceso del servidor proporciona esta información con una DescribePlayerSessions() llamada a Amazon GameLift.

Propiedades	Descripción
GameSessionId	Un identificador único de la sesión de juegos. Use este parámetro para solicitar todas las sesiones de jugador de la sesión de juego especificada. El formato de ID de sesión de juego es `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`. El `GameSessionID` es una cadena de ID personalizada o una Tipo: `std::string` Obligatorio: no
PlayerSessionId	El identificador único de una sesión de jugador. Utilice este parámetro para solicitar una sesión de jugador específica. Tipo: `std::string` Obligatorio: no
PlayerId	El identificador único de un jugador. Utilice este parámetro para solicitar todas las sesiones de jugador para un jugador específico. Consulte Generación de ID de jugador. Tipo: `std::string` Obligatorio: no
PlayerSessionStatusFilter	El estado de la sesión de jugador para filtrar los resultados. Los posibles estados de sesión de jugador son los siguientes: • RESERVADO: se ha recibido la solicitud de sesión de jugador, pero el jugador aún no se

Propiedades	Descripción
	ha conectado al proceso del servidor o aún no se ha validado. • ACTIVO: el proceso del servidor ha validado el jugador y está conectado. • COMPLETO: la conexión del jugador se ha caído. • TIEMPO DE ESPERA AGOTADO: se ha recibido una solicitud de sesión de jugador, pero el jugador no se ha conectado y/o no se ha validado en el plazo de tiempo de espera (60 segundos). Tipo: `std::string` Obligatorio: no
NextToken	El token que indica el inicio de la siguiente página de resultados. Para especificar el inicio del conjunto de resultados, no indique ningún valor. Si se especifica un ID de sesión de jugador, este parámetro se ignora. Tipo: `std::string` Obligatorio: no
Límite	El número máximo de resultados que devolver. Si se especifica un ID de sesión de jugador, este parámetro se ignora. Tipo: `int` Obligatorio: no

StopMatchBackfillRequest

Información utilizada para cancelar una solicitud de reposición de emparejamiento. El servidor del juego comunica esta información al GameLift servicio de Amazon en una StopMatchBackfill() llamada.

Propiedades	Descripción
GameSessionArn	Un identificador único de sesión de juego de la solicitud que se va a cancelar. Tipo: `char[]` Obligatorio: no
MatchmakingConfigurationArn	Un identificador único del emparejador al que se envió esta solicitud. Tipo: `char[]` Obligatorio: no
TicketId	Un identificador único del ticket de solicitud de reposición que se va a cancelar. Tipo: `char[]` Obligatorio: no

AttributeValue

Utilice estos valores en pares de clave-valor de atributo Jugador. Este objeto le permite especificar un valor de atributo mediante cualquiera de los tipos de datos válidos: cadena, número, matriz de cadenas o mapa de datos. Cada objeto `AttributeValue` debe utilizar exactamente una de las propiedades disponibles: `S`, `N`, `SL` o `SDM`.

Propiedades	Descripción
AttrType	Especifica el tipo de valor del atributo. Entre los posibles tipos de valores de atributo se incluyen: • NONE • STRING • DOUBLE • STRING_LIST • STRING_DOUBLE_MAP Obligatorio: no
S	Representa un valor de atributo de cadena. Tipo: `std::string` Obligatorio: no
N	Representa un valor de atributo numérico. Tipo: `double` Obligatorio: no
SL	Representa una matriz de valores de atributos de cadena. Tipo: `std::vector<std::string>` Obligatorio: no
SDM	Representa un diccionario de claves de cadena y valores dobles. Tipo: `std::map<std::string, double>` Obligatorio: no

GetFleetRoleCredentialsRequest

Este tipo de datos proporciona al servidor de juegos un acceso limitado a los otros recursos de AWS. Para obtener más información, consulte Configurar un rol de servicio de IAM para Amazon GameLift.

Propiedades	Descripción
RoleArn	El nombre de recurso de Amazon (ARN) del rol de servicio que amplía el acceso limitado a sus recursos de AWS. Tipo: `std::string` Obligatorio: no
RoleSessionName	El nombre de sesión del rol que puedes usar para identificar de forma exclusiva una AWS Security Token Service AssumeRolesesión. Este nombre aparece en los registros de auditoría, como los de CloudTrail. Tipo: `std::string` Obligatorio: no

AwsLongOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Resultado	El resultado de la acción. Tipo: `long` Obligatorio: no
ResultWithOwnership	El resultado de la acción, expresado como un valor r, para que el código de llamada pueda tomar posesión del objeto.

Propiedades	Descripción
	Tipo: `long&&` Obligatorio: no
Success	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

AwsStringOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Resultado	El resultado de la acción. Tipo: `std::string` Obligatorio: no
ResultWithOwnership	El resultado de la acción, expresado como un valor r, para que el código de llamada pueda tomar posesión del objeto. Tipo: `long&&` Obligatorio: no
Success	Si la acción se realizó correctamente o no.

Propiedades	Descripción
	Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

DescribePlayerSessionsOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Resultado	El resultado de la acción. Tipo: the section called “DescribePlayerSessionsResult” Obligatorio: no
ResultWithOwnership	El resultado de la acción, expresado como un valor r, para que el código de llamada pueda tomar posesión del objeto. Tipo: `Aws::GameLift::Server::Model::DescribePlayerSessionsResult&&` Obligatorio: no
Success	Si la acción se realizó correctamente o no. Tipo: `bool`

Propiedades	Descripción
	Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

DescribePlayerSessionsResult

Una colección de objetos que contiene propiedades para cada sesión de jugador que se empareja con la solicitud.

Propiedades	Descripción
NextToken	Un token que indica el inicio de la siguiente página de resultados secuencial. Utilice el token devuelto con una llamada anterior a esta operación. Para especificar el inicio del conjunto de resultados, no especifique ningún valor. Si se especifica un ID de sesión de jugador, este parámetro se ignora. Tipo: `std::string` Obligatorio: sí
PlayerSessions	Tipo: `IList<`the section called “PlayerSession” `>` Obligatorio:
ResultWithOwnership	El resultado de la acción, expresado como un valor r, para que el código de llamada pueda tomar posesión del objeto. Tipo: `std::string&&`

Propiedades	Descripción
	Obligatorio: no
Success	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

GenericOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Success	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

GenericOutcomeCallable

Este tipo de datos es un resultado genérico asíncrono. Tiene las siguientes propiedades:

Propiedades	Descripción
Success	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

PlayerSession

Este tipo de datos representa la sesión de un jugador que Amazon GameLift pasa al servidor del juego. Para obtener más información, consulte PlayerSession.

Propiedades	Descripción
CreationTime	Tipo: `long` Obligatorio: no
FleetId	Tipo: `std::string` Obligatorio: no
GameSessionId	Tipo: `std::string` Obligatorio: no
IpAddress	Tipo: `std::string` Obligatorio: no
PlayerData	Tipo: `std::string`

Propiedades	Descripción
	Obligatorio: no
PlayerId	Tipo: `std::string` Obligatorio: no
PlayerSessionId	Tipo: `std::string` Obligatorio: no
Puerto	Tipo: `int` Obligatorio: no
Status	Estado de la sesión de juego para filtrar los resultados. Cuando PlayerId se proporciona una PlayerSessionId o, no PlayerSessionStatusFilter tiene ningún efecto en la respuesta. Tipo: una enumeración de `PlayerSessionStatus` . Entre los valores posibles se incluyen: • ACTIVO • COMPLETED • NOT_SET • RESERVED • TIEMPO DE ESPERA AGOTADO Obligatorio: no
TerminationTime	Tipo: `long` Obligatorio: no

Propiedades	Descripción
DnsName	Tipo: `std::string` Obligatorio: no

StartMatchBackfillOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Resultado	El resultado de la acción. Tipo: the section called “StartMatchBackfillResult” Obligatorio: no
ResultWithOwnership	El resultado de la acción, expresado como un valor r, para que el código de llamada pueda tomar posesión del objeto. Tipo: `StartMatchBackfillResult&&` Obligatorio: no
Success	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

StartMatchBackfillResult

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
TicketId	Un identificador único de un ticket de emparejamiento Si aquí no se especific a ningún identificador de billete, Amazon GameLift generará uno en forma de UUID. Utilice este identificador para realizar un seguimiento del estado del ticket de relleno del emparejamiento y recuperar los resultados del emparejamiento. Tipo: `std::string` Obligatorio: no

GetComputeCertificateOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Resultado	El resultado de la acción. Tipo: the section called “GetComputeCertificateResult” Obligatorio: no
ResultWithOwnership	El resultado de la acción, expresado como un valor r, para que el código de llamada pueda tomar posesión del objeto. Tipo: `Aws::GameLift::Server::Model::GetComputeCertificateResult&&`

Propiedades	Descripción
	Obligatorio: no
Success	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

GetComputeCertificateResult

La ruta al certificado TLS de su recurso informático y el nombre de host del equipo.

Propiedades	Descripción
CertificatePath	La ruta al certificado TLS de su recurso informático. Cuando se utiliza una flota GameLift gestionada por Amazon, esta ruta contiene: • `certificate.pem` : el certificado del usuario final. La cadena de certificados completa es la combinación del `certificateChain.pem` adjunto a este certificado. • `certificateChain.pem` : la cadena de certificados que contiene el certificado raíz y los certificados intermedios. • `rootCertificate.pem` : el certificado raíz.

Propiedades	Descripción
	• `privateKey.pem` : la clave privada del certificado del usuario final. Tipo: `std::string` Obligatorio: no
ComputeName	El nombre del recurso informático. Tipo: `std::string` Obligatorio: no

GetFleetRoleCredentialsOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Resultado	El resultado de la acción. Tipo: the section called “GetFleetRoleCredentialsResult” Obligatorio: no
ResultWithOwnership	El resultado de la acción, expresado como un valor r, para que el código de llamada pueda tomar posesión del objeto. Tipo: `Aws::GameLift::Server::Model::GetFleetRoleCredentialsResult` Obligatorio: no
Success	Si la acción se realizó correctamente o no.

Propiedades	Descripción
	Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

GetFleetRoleCredentialsResult

Propiedades	Descripción
AccessKeyId	El ID de la clave de acceso para autenticar y proporcionar acceso a los recursos de AWS. Tipo: `string` Obligatorio: no
AssumedRoleId	El ID del usuario al que pertenece el rol de servicio. Tipo: `string` Obligatorio: no
AssumedRoleUserArn	El nombre de recurso de Amazon (ARN) del usuario al que pertenece el rol de servicio. Tipo: `string` Obligatorio: no
Expiration	El tiempo que queda hasta que caduquen las credenciales de la sesión.

Propiedades	Descripción
	Tipo: `DateTime` Obligatorio: no
SecretAccessKey	El ID de clave de acceso secreta para la autenticación. Tipo: `string` Obligatorio: no
SessionToken	Un token para identificar la sesión activa actual que interactúa con los recursos de AWS. Tipo: `string` Obligatorio: no
Success	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

InitSDKOutcome

> **Note**
>
> `InitSDKOutcome` se devuelve solo cuando compila el SDK con la marca `std`. Si realiza la compilación con la marca `nostd`, se devuelve the section called “GenericOutcome” en su lugar.

Propiedades	Descripción
Success	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

GameLiftError

Propiedades	Descripción
ErrorType	Tipo de error. Tipo: una enumeración de `GameLiftErrorType` . Obligatorio: no
ErrorName	El nombre del error. Tipo: `std::string` Obligatorio: no
ErrorMessage	Mensaje de error. Tipo: `std::string` Obligatorio: no

Enums

Las enumeraciones definidas para el SDK GameLift del servidor Amazon (C++) se definen de la siguiente manera:

GameLiftErrorType

Valor de cadena que indica el tipo de error. Los valores válidos son:

- BAD_REQUEST_EXCEPTION
- GAMESESSION_ID_NOT_SET: no se ha establecido el ID de sesión de juego.
- INTERNAL_SERVICE_EXCEPTION
- LOCAL_CONNECTION_FAILED — Falló la conexión local con Amazon. GameLift
- NETWORK_NOT_INITIALIZED: la red no se ha inicializado.
- SERVICE_CALL_FAILED: se ha producido un error en la llamada a un servicio de AWS.
- WEBSOCKET_CONNECT_FAILURE
- WEBSOCKET_CONNECT_FAILURE_FORBIDDEN
- WEBSOCKET_CONNECT_FAILURE_INVALID_URL
- WEBSOCKET_CONNECT_FAILURE_TIMEOUT
- ALREADY_INITIALIZED: el GameLift servidor o cliente de Amazon ya se inicializó con Initialize ().
- FLEET_MISMATCH: la flota de destino no coincide con la flota de gameSession o playerSession.
- GAMELIFT_CLIENT_NOT_INITIALIZED — El cliente de Amazon no se ha inicializado. GameLift
- GAMELIFT_SERVER_NOT_INITIALIZED — El servidor de Amazon no se ha inicializado. GameLift
- GAME_SESSION_ENDED_FAILED: el SDK de GameLift Amazon Server no pudo contactar con el servicio para informar que la sesión de juego había finalizado.
- GAME_SESSION_NOT_READY: no se activó la sesión de juego del servidor GameLift Amazon.
- GAME_SESSION_READY_FAILED: el SDK de GameLift Amazon Server no pudo contactar con el servicio para informar que la sesión de juego estaba lista.
- INITIALIZATION_MISMATCH: se llamó a un método de cliente después de Server::Initialize() o viceversa.

- NOT_INITIALIZED — El GameLift servidor o cliente de Amazon no se ha inicializado con Initialize ().
- NO_TARGET_ALIASID_SET: no se ha establecido un aliasId de destino.
- NO_TARGET_FLEET_SET: no se ha establecido una flota de destino.
- PROCESS_ENDING_FAILED: el SDK de GameLift Amazon Server no pudo contactar con el servicio para informar que el proceso está finalizando.
- PROCESS_NOT_ACTIVE — El proceso del servidor aún no está activo, no está vinculado a a y no puede aceptarlo ni procesarlo. GameSession PlayerSessions
- PROCESS_NOT_READY: el proceso de servidor aún no está listo para activarse.
- PROCESS_READY_FAILED: el SDK de Amazon GameLift Server no pudo contactar con el servicio para informar que el proceso está listo.
- SDK_VERSION_DETECTION_FAILED: no se pudo detectar la versión del SDK.
- STX_CALL_FAILED: no se pudo realizar una llamada al componente de backend del servidor xSTx.
- STX_INITIALIZATION_FAILED: no se pudo inicializar el componente de backend del servidor xSTx.
- UNEXPECTED_PLAYER_SESSION: el servidor ha detectado una sesión de jugador no registrada.
- WEBSOCKET_CONNECT_FAILURE
- WEBSOCKET_CONNECT_FAILURE_FORBIDDEN
- WEBSOCKET_CONNECT_FAILURE_INVALID_URL
- WEBSOCKET_CONNECT_FAILURE_TIMEOUT
- WEBSOCKET_RETRIABLE_SEND_MESSAGE_FAILURE: error recuperable al enviar un mensaje al servicio. GameLift WebSocket
- WEBSOCKET_SEND_MESSAGE_FAILURE: error al enviar un mensaje al GameLift servicio. WebSocket
- MATCH_BACKFILL_REQUEST_VALIDATION: se ha producido un error en la validación de la solicitud.
- PLAYER_SESSION_REQUEST_VALIDATION: se ha producido un error en la validación de la solicitud.

PlayerSessionCreationPolicy

Valor de cadena que indica si la sesión de juego acepta jugadores nuevos. Los valores válidos son:

- ACCEPT_ALL: se aceptan todas las sesiones de jugador nuevas.
- DENY_ALL: se rechazan todas las sesiones de jugador nuevas.
- NOT_SET: la sesión de juego no está configurada para aceptar o denegar sesiones de nuevos jugadores.

Referencia del SDK del servidor C++ de Amazon GameLift 3.x

Puede utilizar esta referencia del SDK del servidor C++ de Amazon GameLift 3.x como ayuda para preparar su juego multijugador para su uso con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

Temas

- Referencia del SDK del servidor de Amazon GameLift (C++): Acciones
- Referencia del SDK del servidor de Amazon GameLift (C++): Tipos de datos

Referencia del SDK del servidor de Amazon GameLift (C++): Acciones

Puede utilizar esta referencia del SDK del servidor C++ de Amazon GameLift como ayuda para preparar su juego multijugador para su uso con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

Acciones

- AcceptPlayerSession()
- ActivateGameSession()
- DescribePlayerSessions()
- GetGameSessionId()
- GetInstanceCertificate()
- GetSdkVersion()
- GetTerminationTime()
- InitSDK()

- ProcessEnding()
- ProcessReady()
- ProcessReadyAsync()
- RemovePlayerSession()
- StartMatchBackfill()
- StopMatchBackfill()
- TerminateGameSession()
- UpdatePlayerSessionCreationPolicy()
- Destroy()

AcceptPlayerSession()

Notifica al servicio de Amazon GameLift que un jugador con el ID de sesión de jugador especificado se ha conectado al proceso del servidor y requiere validación. Amazon GameLift verifica que el ID de sesión del jugador es válido, es decir, que el ID de jugador ha reservado una ranura de jugador en la sesión de juego. Una vez completada la validación, Amazon GameLift cambia el estado de la ranura de jugador de RESERVADO a ACTIVO.

Sintaxis

```
GenericOutcome AcceptPlayerSession(const std::string& playerSessionId);
```

Parámetros

playerSessionId

ID único emitido por el servicio de Amazon GameLift como respuesta a una llamada a la acción CreatePlayerSession de la API de Amazon GameLift del SDK de AWS. El cliente de juego hace referencia a este ID cuando se conecta al proceso del servidor.

Tipo: std::string

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo muestra una función para gestionar una solicitud de conexión, incluida la validación y el rechazo de ID de sesión de jugador no válidos.

```
void ReceiveConnectingPlayerSessionID (Connection& connection, const std::string&
 playerSessionId){
    Aws::GameLift::GenericOutcome connectOutcome =
        Aws::GameLift::Server::AcceptPlayerSession(playerSessionId);
    if(connectOutcome.IsSuccess())
    {
        connectionToSessionMap.emplace(connection, playerSessionId);
        connection.Accept();
    }
    else
    {
        connection.Reject(connectOutcome.GetError().GetMessage();
    }
}
```

ActivateGameSession()

Notifica al servicio de Amazon GameLift que el proceso del servidor ha iniciado una sesión de juego y que está listo para recibir las conexiones de los jugadores. Esta acción debe llamarse como parte de la función de devolución de llamada `onStartGameSession()`, después de completar la inicialización de todas las sesiones de juego.

Sintaxis

```
GenericOutcome ActivateGameSession();
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo muestra cómo se llama a `ActivateGameSession()` como parte de la función de devolución de llamada `onStartGameSession()`.

```
void onStartGameSession(Aws::GameLift::Model::GameSession myGameSession)
{
   // game-specific tasks when starting a new game session, such as loading map
   GenericOutcome outcome = Aws::GameLift::Server::ActivateGameSession();
}
```

DescribePlayerSessions()

Recupera datos de sesión de jugador, incluida la configuración, los metadatos de la sesión y los datos de jugador. Utilice esta acción para obtener información para una única sesión de jugador, para todas las sesiones de jugador de una sesión de juego o para todas las sesiones de jugador asociadas a un solo ID de jugador.

Sintaxis

```
DescribePlayerSessionsOutcome DescribePlayerSessions (
    const Aws::GameLift::Server::Model::DescribePlayerSessionsRequest
 &describePlayerSessionsRequest);
```

Parámetros

describePlayerSessionsRequest

Es un objeto DescribePlayerSessionsRequest que describe las sesiones de jugador a recuperar.

Obligatorio: sí

Valor devuelto

Si funciona correctamente, devuelve un objeto `DescribePlayerSessionsOutcome` que contiene un conjunto de objetos de sesión de jugador que se ajusta a los parámetros de la solicitud. Los objetos de las sesiones de jugador tienen una estructura idéntica al tipo de datos PlayerSession de la API de Amazon GameLift del SDK de AWS.

Ejemplo

Este ejemplo muestra una solicitud de todas las sesiones de jugador conectadas activamente a una sesión de juego específica. Al omitir `NextToken` y definir el valor `Limit` en 10, Amazon GameLift devuelve los primeros 10 registros de sesiones de jugador que coincidan con la solicitud.

```
// Set request parameters
```

```
Aws::GameLift::Server::Model::DescribePlayerSessionsRequest request;
request.SetPlayerSessionStatusFilter(Aws::GameLift::Server::Model::PlayerSessionStatusMapper::G
request.SetLimit(10);
request.SetGameSessionId("the game session ID");    // can use GetGameSessionId()

// Call DescribePlayerSessions
Aws::GameLift::DescribePlayerSessionsOutcome playerSessionsOutcome =
    Aws::GameLift::Server::DescribePlayerSessions(request);
```

GetGameSessionId()

Recupera un identificador único de la sesión de juego alojada actualmente por el proceso del servidor, siempre que esté activo. El identificador se devuelve en el formato de ARN: `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`.

En el caso de los procesos inactivos que no activados aún con una sesión de juego, la llamada devuelve `Success=True` y `GameSessionId=""` (una cadena vacía).

Sintaxis

```
AwsStringOutcome GetGameSessionId();
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Si funciona correctamente, devuelve el ID de sesión del juego como objeto `AwsStringOutcome`. Si no funciona, devuelve un mensaje de error.

Ejemplo

```
Aws::GameLift::AwsStringOutcome sessionIdOutcome =
    Aws::GameLift::Server::GetGameSessionId();
```

GetInstanceCertificate()

Recupera la ubicación del archivo de un certificado TLS codificado con PEM que está asociado a la flota y sus instancias. AWS Certificate Manager genera este certificado al crear una nueva flota con la configuración del certificado establecida en GENERADO. Utilice este certificado para establecer una conexión segura con un cliente de juego y para cifrar la comunicación cliente/servidor.

Sintaxis

```
GetInstanceCertificateOutcome GetInstanceCertificate();
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Si se ejecuta correctamente, devuelve un objeto `GetInstanceCertificateOutcome` que contiene la ubicación del archivo de certificado TLS y la cadena de certificados de la flota, que se almacena en la instancia. En la instancia también se almacena un archivo de certificado raíz extraído de la cadena de certificados. Si no funciona, devuelve un mensaje de error.

Para obtener más información sobre el certificado y los datos de la cadena de certificados, consulte Elementos de respuesta de GetCertificate en la referencia de la API de AWS Certificate Manager.

Ejemplo

```
Aws::GameLift::GetInstanceCertificateOutcome certificateOutcome =
    Aws::GameLift::Server::GetInstanceCertificate();
```

GetSdkVersion()

Devuelve el número de versión actual del SDK en uso.

Sintaxis

```
AwsStringOutcome GetSdkVersion();
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Si funciona correctamente, devuelve la versión del SDK actual como objeto `AwsStringOutcome`. La cadena devuelta solo incluye el número de versión (por ejemplo, («3.1.5»). Si no funciona, devuelve un mensaje de error.

Ejemplo

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =
    Aws::GameLift::Server::GetSdkVersion();
```

GetTerminationTime()

Devuelve la hora a la que está programada el cierre de un proceso de servidor, si hay una hora de terminación disponible. Un proceso de servidor realiza esta acción después de recibir una devolución de llamada onProcessTerminate() desde el servicio de Amazon GameLift. Amazon GameLift puede llamar a onProcessTerminate() por los siguientes motivos: (1) cuando el proceso del servidor ha informado de un estado deficiente o no ha respondido a Amazon GameLift, (2) cuando se termina la instancia durante un evento de reducción vertical o (3) cuando una instancia se cierra debido a una interrupción de spot.

Si el proceso ha recibido una devolución de llamada onProcessTerminate(), el devuelto es la hora de terminación estimada. Si el proceso no ha recibido ninguna devolución de llamada onProcessTerminate(), se devuelve un mensaje de error. Más información acerca del apagado de un proceso de servidor.

Sintaxis

```
AwsLongOutcome GetTerminationTime();
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Si el proceso se realiza correctamente, devuelve la hora de terminación como un objeto AwsLongOutcome. El valor es la hora de terminación expresado en ciclos transcurridos desde 0001 00:00:00. Por ejemplo, el valor de fecha y hora 13-09-2020 12:26:40 -000Z es igual a 637355968000000000 ciclos. Si no hay una hora de terminación disponible, devuelve un mensaje de error.

Ejemplo

```
Aws::GameLift::AwsLongOutcome TermTimeOutcome =
    Aws::GameLift::Server::GetTerminationTime();
```

InitSDK()

Inicializa el SDK de Amazon GameLift. Este método debe llamarse en el momento del lanzamiento, antes de cualquier otra inicialización relacionada con Amazon GameLift.

Sintaxis

```
InitSDKOutcome InitSDK();
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Si funciona correctamente, devuelve un objeto InitSdkOutcome e indica que el proceso del servidor está listo para llamar a ProcessReady().

Ejemplo

```
Aws::GameLift::Server::InitSDKOutcome initOutcome =
    Aws::GameLift::Server::InitSDK();
```

ProcessEnding()

Informa al servicio de Amazon GameLift de que el proceso del servidor se va a detener. Este método debe llamarse después de realizar las demás tareas de limpieza, incluido el cierre de todas las sesiones de juego activas. Se debe salir de este método con un código de salida de 0; un código de salida que no sea 0 provoca un mensaje de evento que afirma que no se ha salido del proceso correctamente.

Después de que el método finalice con un código de 0, puede terminar el proceso con un código de salida correcto. También puede salir del proceso con un código de error. Si sale con un código de error, el evento de la flota indicará que el proceso ha finalizado incorrectamente (`SERVER_PROCESS_TERMINATED_UNHEALTHY`).

Sintaxis

```
GenericOutcome ProcessEnding();
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
Aws::GameLift::GenericOutcome outcome = Aws::GameLift::Server::ProcessEnding();
if (outcome.Success)
    exit(0);  // exit with success
// otherwise, exit with error code
exit(errorCode);
```

ProcessReady()

Notifica al servicio de Amazon GameLift que el proceso del servidor está listo para alojar sesiones de juego. Llame a este método después de invocar correctamente a InitSDK() y completar las tareas de configuración necesarias antes de que el proceso del servidor pueda alojar una sesión de juego. Se debe llamar a este método solo una vez por proceso.

La llamada es síncrona. Para realizar una llamada asíncrona, utilice ProcessReadyAsync(). Consulte Inicialización del proceso del servidor para obtener más detalles.

Sintaxis

```
GenericOutcome ProcessReady(
    const Aws::GameLift::Server::ProcessParameters &processParameters);
```

Parámetros

processParameters

Es un objeto ProcessParameters que comunica la siguiente información acerca del proceso del servidor:

- Nombres de métodos de devolución de llamada, implementados en el código de servidor de juegos, que el servicio de Amazon GameLift invoca para comunicarse con el proceso del servidor.

- Número de puerto de escucha del servidor de proceso.
- Ruta a cualquier archivo específico de la sesión de juego que desea que Amazon GameLift capture y almacene.

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo ilustra las implementaciones tanto de la función de llamada como de la función de devolución de llamada ProcessReady().

```
// Set parameters and call ProcessReady
std::string serverLog("serverOut.log");        // Example of a log file written by the
 game server
std::vector<std::string> logPaths;
logPaths.push_back(serverLog);

int listenPort = 9339;

Aws::GameLift::Server::ProcessParameters processReadyParameter =
 Aws::GameLift::Server::ProcessParameters(
    std::bind(&Server::onStartGameSession, this, std::placeholders::_1),
    std::bind(&Server::onProcessTerminate, this),
    std::bind(&Server::OnHealthCheck, this),
    std::bind(&Server::OnUpdateGameSession, this),
    listenPort,
    Aws::GameLift::Server::LogParameters(logPaths));

Aws::GameLift::GenericOutcome outcome =
   Aws::GameLift::Server::ProcessReady(processReadyParameter);

// Implement callback functions
void Server::onStartGameSession(Aws::GameLift::Model::GameSession myGameSession)
{
   // game-specific tasks when starting a new game session, such as loading map
   GenericOutcome outcome =
       Aws::GameLift::Server::ActivateGameSession (maxPlayers);
}
```

```
void Server::onProcessTerminate()
{
   // game-specific tasks required to gracefully shut down a game session,
   // such as notifying players, preserving game state data, and other cleanup
   GenericOutcome outcome = Aws::GameLift::Server::ProcessEnding();
}

bool Server::onHealthCheck()
{
    bool health;
    // complete health evaluation within 60 seconds and set health
    return health;
}
```

ProcessReadyAsync()

Notifica al servicio de Amazon GameLift que el proceso del servidor está listo para alojar sesiones de juego. Este método debe llamarse cuando el proceso del servidor esté listo para alojar una sesión de juego. Los parámetros especifican los nombres de las funciones de devolución de llamada a las que Amazon GameLift llama en determinadas circunstancias. El código de servidor de juegos debe implementar estas funciones.

La llamada es asíncrona. Para realizar una llamada síncrona, utilice ProcessReady(). Consulte Inicialización del proceso del servidor para obtener más detalles.

Sintaxis

```
GenericOutcomeCallable ProcessReadyAsync(
    const Aws::GameLift::Server::ProcessParameters &processParameters);
```

Parámetros

processParameters

Es un objeto ProcessParameters que comunica la siguiente información acerca del proceso del servidor:

- Nombres de métodos de devolución de llamada, implementados en el código de servidor de juegos, que el servicio de Amazon GameLift invoca para comunicarse con el proceso del servidor.

- Número de puerto de escucha del servidor de proceso.
- Ruta a cualquier archivo específico de la sesión de juego que desea que Amazon GameLift capture y almacene.

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
// 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()

Informa al servicio de Amazon GameLift de que un jugador con el ID de sesión de jugador especificado se ha desconectado del proceso del servidor. Como respuesta, Amazon GameLift cambia el estado de la ranura de jugador a disponible, por lo que se le puede asignar un jugador nuevo.

Sintaxis

```
GenericOutcome RemovePlayerSession(
    const std::string& playerSessionId);
```

Parámetros

playerSessionId

ID único emitido por el servicio de Amazon GameLift como respuesta a una llamada a la acción CreatePlayerSession de la API de Amazon GameLift del SDK de AWS. El cliente de juego hace referencia a este ID cuando se conecta al proceso del servidor.

Tipo: std::string

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
Aws::GameLift::GenericOutcome disconnectOutcome =
```

```
    Aws::GameLift::Server::RemovePlayerSession(playerSessionId);
```

StartMatchBackfill()

Envía una solicitud para encontrar nuevos jugadores para ranuras abiertas en una sesión de juego creada con FlexMatch. Consulte también la acción del SDK de AWS StartMatchBackfill(). Con esta acción, un proceso del servidor de juegos que aloja la sesión de juego puede iniciar solicitudes de reposición de emparejamiento. Puede obtener más información sobre la característica de reposición de FlexMatch.

Esta acción es asíncrona. Si se emparejan correctamente nuevos jugadores, el servicio de Amazon GameLift ofrece datos actualizados del emparejador si se invoca la función de devolución de llamada `OnUpdateGameSession()`.

Un proceso del servidor solo puede tener una solicitud de reposición de emparejamiento activa a la vez. Para enviar una nueva solicitud, en primer lugar llame a StopMatchBackfill() para cancelar la solicitud original.

Sintaxis

```
StartMatchBackfillOutcome StartMatchBackfill (
    const Aws::GameLift::Server::Model::StartMatchBackfillRequest
 &startBackfillRequest);
```

Parámetros

StartMatchBackfillRequest

Un objeto StartMatchBackfillRequest que comunica la siguiente información:

- Un ID de ticket que se asignará a la solicitud de reposición. Esta información es opcional; si no se proporciona ningún ID, Amazon GameLift generará automáticamente uno.
- El creador de emparejamientos al que se enviará la solicitud. El ARN de configuración completo es obligatorio. Este valor se puede obtener de los datos del creador de emparejamientos de la sesión de juego.
- El ID de la sesión de juego que está en fase de reposición.
- Datos de emparejamiento disponibles para los jugadores actuales de la sesión de juego.

Obligatorio: sí

Valor devuelto

Devuelve un objeto StartMatchBackfillOutcome con el ticket de reposición de emparejamiento o un error con un mensaje de error. Podrá realizar el seguimiento del estado del ticket con la acción del SDK de AWS DescribeMatchmaking().

Ejemplo

```
// Build a backfill request
std::vector<Player> players;
Aws::GameLift::Server::Model::StartMatchBackfillRequest startBackfillRequest;
startBackfillRequest.SetTicketId("a ticket ID");
  //optional, autogenerated if not provided
startBackfillRequest.SetMatchmakingConfigurationArn("the matchmaker configuration
 ARN"); //from the game session matchmaker data
startBackfillRequest.SetGameSessionArn("the game session ARN");
  // can use GetGameSessionId()
startBackfillRequest.SetPlayers(players);
    //from the game session matchmaker data

// Send backfill request
Aws::GameLift::StartMatchBackfillOutcome backfillOutcome =
    Aws::GameLift::Server::StartMatchBackfill(startBackfillRequest);

// Implement callback function for backfill
void Server::OnUpdateGameSession(Aws::GameLift::Server::Model::GameSession gameSession,
 Aws::GameLift::Server::Model::UpdateReason updateReason, std::string backfillTicketId)
{
   // handle status messages
   // perform game-specific tasks to prep for newly matched players
}
```

StopMatchBackfill()

Cancela una solicitud de reposición de emparejamiento activa que se creó con StartMatchBackfill(). Consulte también la acción del SDK de AWS StopMatchmaking(). Puede obtener más información sobre la característica de reposición de FlexMatch.

Sintaxis

```
GenericOutcome StopMatchBackfill (
    const Aws::GameLift::Server::Model::StopMatchBackfillRequest &stopBackfillRequest);
```

Parámetros

StopMatchBackfillRequest

Un objeto StopMatchBackfillRequest que identifica el ticket de emparejamiento que se va a cancelar:

- ID del ticket asignado a la solicitud de reposición que se va a cancelar
- el creador de emparejamientos al que se envió la solicitud de reposición
- sesión de juego asociada a la solicitud de reposición

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
// 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()

Este método está obsoleto con la versión 4.0.1. En su lugar, el proceso del servidor debería llamar a ProcessEnding() una vez finalizada la sesión de juego.

Informa al servicio de Amazon GameLift de que el proceso del servidor ha finalizado la sesión de juego actual. Se llama a esta acción cuando el proceso del servidor permanece activo y listo para alojar una nueva sesión de juego. Solo debe llamarse una vez finalizado el procedimiento de finalización de la sesión de juego, ya que indica a Amazon GameLift que el proceso del servidor está disponible inmediatamente para alojar una nueva sesión de juego.

No se llamará a esta acción si el proceso del servidor se interrumpe una vez finalizada la sesión de juego. En su lugar, se llamará a ProcessEnding() para indicar que tanto la sesión de juego como el proceso del servidor están finalizando.

Sintaxis

```
GenericOutcome TerminateGameSession();
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

UpdatePlayerSessionCreationPolicy()

Actualiza la capacidad de la sesión de juego actual para aceptar sesiones de jugador nuevas. Una sesión de juego se puede configurar para que acepte o deniegue todas las sesiones nuevas de los jugadores. Consulte también la acción del SDK de AWS UpdateGameSession().

Sintaxis

```
GenericOutcome UpdatePlayerSessionCreationPolicy(
    Aws::GameLift::Model::PlayerSessionCreationPolicy newPlayerSessionPolicy);
```

Parámetros

newPlayerSessionPolicy

Valor de cadena que indica si la sesión de juego acepta jugadores nuevos.

Tipo: enum Aws::GameLift::Model::PlayerSessionCreationPolicy. Los valores válidos son:

- ACCEPT_ALL: se aceptan todas las sesiones de jugador nuevas.
- DENY_ALL: se rechazan todas las sesiones de jugador nuevas.

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo establece la política de participación en la sesión de juego actual para aceptar todos los jugadores.

```
Aws::GameLift::GenericOutcome outcome =
 Aws::GameLift::Server::UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::PlayerSessionCr
```

Destroy()

Limpia la memoria asignada por initSDK() durante la inicialización del servidor de juegos. Utilice este método después de finalizar un proceso de servidor de juegos para evitar desperdiciar memoria del servidor.

Sintaxis

```
GenericOutcome Aws::GameLift::Server::Destroy();
```

Parámetros

No hay parámetros.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

En este ejemplo, se limpia la memoria asignada por InitSDK una vez finalizado el proceso del servidor de juegos.

```
if (Aws::GameLift::Server::ProcessEnding().IsSuccess()) {
  Aws::GameLift::Server::Destroy();
  exit(0);
}
```

Referencia del SDK del servidor de Amazon GameLift (C++): Tipos de datos

Puede utilizar esta referencia del SDK del servidor C++ de Amazon GameLift como ayuda para preparar su juego multijugador para su uso con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

Esta API se define en `GameLiftServerAPI.h`, `LogParameters.h` y `ProcessParameters.h`.

- Acciones
- Tipos de datos

DescribePlayerSessionsRequest

Este tipo de datos se utiliza para especificar qué sesión o sesiones de jugador recuperar. Puede utilizarlo para las siguientes tareas:

- Proporcionar un PlayerSessionId para solicitar una sesión de jugador específica.
- Proporcionar un GameSessionId para solicitar todas las sesiones de jugador de la sesión de juego especificada.
- Proporcionar un PlayerId para solicitar todas las sesiones de jugador del jugador especificado.

Para grandes conjuntos de sesiones de jugador, utilice los parámetros de paginación para recuperar los resultados en bloques consecutivos.

Contenido

GameSessionId

Identificador único de la sesión de juego. Use este parámetro para solicitar todas las sesiones de jugador de la sesión de juego especificada. El formato de ID de la sesión de juego es el siguiente: `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`. El valor de la **<**cadena de ID**>** es una cadena de ID personalizada o (si se especificó una cuando se creó la sesión de juego) una cadena generada.

Tipo: String

Requerido: No

Límite

Número máximo de resultados a devolver. Use este parámetro con NextToken para obtener resultados en un conjunto de páginas secuenciales. Si se especifica un ID de sesión de jugador, este parámetro se ignora.

Tipo: entero

Obligatorio: no

NextToken

Token que indica el inicio de la siguiente página de resultados secuencial. Utilice el token devuelto con una llamada anterior a esta acción. Para especificar el inicio del conjunto de resultados, no indique ningún valor. Si se especifica un ID de sesión de jugador, este parámetro se ignora.

Tipo: String

Requerido: No

PlayerId

Identificador único de un jugador. Los ID de jugador los define el desarrollador. Consulte Generación de ID de jugador.

Tipo: String

Requerido: No

PlayerSessionId

Identificador único de una sesión de jugador.

Tipo: String

Requerido: No

PlayerSessionStatusFilter

Estado de la sesión de juego para filtrar los resultados. Los posibles estados de sesión de jugador son:

- RESERVED: se ha recibido la solicitud de sesión de jugador, pero el jugador aún no se ha conectado al proceso del servidor o aún no se ha validado.
- ACTIVE: el proceso del servidor ha validado el jugador y actualmente está conectado.
- COMPLETED: ha caído la conexión del jugador.
- TIMEDOUT: se ha recibido una solicitud de sesión de jugador, pero el jugador no se ha conectado y/o no se ha validado en el plazo de tiempo de espera (60 segundos).

Tipo: String

Requerido: No

LogParameters

Este tipo de datos se utiliza para identificar los archivos generados durante una sesión de juego que desea que Amazon GameLift cargue y almacene cuando finalice la sesión de juego. Esta información se comunicará al servicio de Amazon GameLift en una llamada a ProcessReady().

Contenido

logPaths

Rutas de directorio a archivos de registro del servidor de juegos que desea que Amazon GameLift almacene para futuros accesos. Estos archivos se generan durante cada sesión de juego. Los nombres y las rutas de los archivos se definen en el servidor de juegos y se almacenan en el directorio raíz de compilación del juego. Las rutas del registro deben ser absolutas. Por ejemplo, si la compilación del juego almacena los logs de sesión de juego en una ruta del tipo `MyGame\sessionlogs\`, entonces la ruta de los logs sería `c:\game\MyGame\sessionLogs` (en una instancia Windows) o `/local/game/MyGame/sessionLogs` (en una instancia Linux).

Tipo: std:vector<std::string>

Obligatorio: no

ProcessParameters

Este tipo de datos contiene el conjunto de parámetros enviado a un servicio de Amazon GameLift en una llamada a ProcessReady().

Contenido

puerto

Es el número de puerto en el que escucha el proceso del servidor para recibir conexiones de jugador nuevas. El valor debe estar en el rango de puertos configurado para cualquier flota que implemente esa compilación de servidor de juegos. Este número de puerto se incluye en los objetos de sesión de juego y de jugador, que las sesiones de juego utilizan a la hora de conectarse a un proceso del servidor.

Tipo: entero

Obligatorio: sí

logParameters

Objeto con una lista de rutas de directorio a archivos de log de la sesión de juego.

Tipo: Aws::GameLift::Server::LogParameters

Obligatorio: no

onStartGameSession

Nombre de la función de devolución de llamada que el servicio de Amazon GameLift invoca para activar una nueva sesión de juego. Amazon GameLift llama a esta función como respuesta a la solicitud de cliente CreateGameSession. La función de devolución de llamada pasa un objeto GameSession (definido en la Referencia de la API del servicio de Amazon GameLift).

Escriba: `const std::function<void(Aws::GameLift::Model::GameSession)> onStartGameSession`

Obligatorio: sí

onProcessTerminate

Nombre de la función de devolución de llamada que el servicio de Amazon GameLift invoca para forzar el cierre del proceso de servidor. Después de llamar a esta función, Amazon GameLift espera cinco minutos hasta que el proceso del servidor se cierre y responda con una llamada a ProcessEnding(). Si no recibe respuesta, cierra el proceso del servidor.

Escriba: `std::function<void()> onProcessTerminate`

Obligatorio: no

onHealthCheck

Nombre de la función de devolución de llamada que invoca el servicio de Amazon GameLift para solicitar un informe de estado del proceso de servidor. Amazon GameLift llama a esta función cada 60 segundos. Después de llamar a esta función, Amazon GameLift espera una respuesta durante 60 segundos y si no recibe ninguna, registra el proceso del servidor como en mal estado.

Escriba: `std::function<bool()> onHealthCheck`

Obligatorio: no

onUpdateGameSession

Nombre de la función de devolución de llamada que el servicio de Amazon GameLift invoca para pasar un objeto de sesión de juego actualizado al proceso de servidor. Amazon GameLift

llama a esta función cuando se ha procesado una solicitud de reposición de emparejamiento para proporcionar datos actualizados de los emparejadores. Pasa un objeto GameSession, una actualización de estado (`updateReason`) y el ID del ticket de reposición de emparejamiento.

Escriba:
`std::function<void(Aws::GameLift::Server::Model::UpdateGameSession)>`
`onUpdateGameSession`

Obligatorio: no

StartMatchBackfillRequest

Este tipo de datos se utiliza para enviar una solicitud de reposición de emparejamiento. La información se comunicará al servicio de Amazon GameLift en una llamada a StartMatchBackfill().

Contenido

GameSessionArn

Identificador único de la sesión de juego. La acción de la API GetGameSessionId() devuelve el identificador en formato de ARN.

Tipo: String

Obligatorio: sí

MatchmakingConfigurationArn

Identificador único, en forma de un ARN, que el creador de emparejamientos utiliza para esta solicitud. Para encontrar el creador de emparejamientos que se usó para crear la sesión de juego original, busque en el objeto de sesión de juego, en la propiedad de datos del creador de emparejamientos. Puede obtener más información sobre los datos del emparejador en Trabajo con datos del emparejador.

Tipo: String

Obligatorio: sí

Players

Un conjunto de datos que representa a todos los jugadores que están actualmente en la sesión de juego. El creador de emparejamientos utiliza esta información para buscar nuevos jugadores

que son idóneos para los jugadores actuales. Para obtener una descripción del formato del objeto Player, consulte Guía de referencia de la API de Amazon GameLift. Para encontrar los atributos, ID y asignaciones de equipo del jugador, busque en el objeto de sesión de juego, en la propiedad de datos del creador de emparejamientos. Si el creador de emparejamientos utiliza latencia, recopile la latencia actualizada para la región actual e inclúyala en los datos de cada jugador.

Tipo: std:vector<player>

Obligatorio: sí

TicketId

Identificador único para un ticket de solicitud de emparejamiento o reposición de emparejamiento. Si no se proporciona ningún valor aquí, Amazon GameLift generará uno en forma de UUID. Use este identificador para realizar un seguimiento del estado del ticket de reposición de emparejamiento o cancelar la solicitud si es necesario.

Tipo: String

Requerido: No

StopMatchBackfillRequest

Este tipo de datos se utiliza para cancelar una solicitud de reposición de emparejamiento. La información se comunicará al servicio de Amazon GameLift en una llamada a StopMatchBackfill().

Contenido

GameSessionArn

Identificador único de sesión de juego asociado a la solicitud que se va a cancelar.

Tipo: String

Obligatorio: sí

MatchmakingConfigurationArn

Identificador único del creador de emparejamientos al que se envió esta solicitud.

Tipo: String

Obligatorio: sí

TicketId

Identificador único del ticket de solicitud de reposición que se va a cancelar.

Tipo: String

Obligatorio: sí

Referencia del SDK del servidor de Amazon GameLift para C#

Puede utilizar esta referencia del SDK del servidor C# de Amazon GameLift como ayuda para preparar su juego multijugador para su uso con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

Temas

- Referencia del SDK del servidor de Amazon GameLift 5.x para C# y Unity
- Referencia del SDK del servidor Amazon GameLift 4.x para C#

Referencia del SDK del servidor de Amazon GameLift 5.x para C# y Unity

Puede utilizar esta referencia del SDK del servidor C# de Amazon GameLift 5.x como ayuda para preparar su juego multijugador para su uso con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos y para obtener información sobre el uso del complemento del SDK del servidor de C# para Unity, consulte Integración de Amazon GameLift en un proyecto de Unity. El SDK del servidor de Amazon GameLift 5.x para C# es compatible con .NET 4.6 y .NET 6.

Temas

- Referencia GameLift del SDK del servidor Amazon para C# y Unity: acciones
- Referencia del SDK del servidor de Amazon GameLift para C# y Unity: Tipos de datos

Referencia GameLift del SDK del servidor Amazon para C# y Unity: acciones

Esta referencia del SDK del servidor GameLift C# de Amazon te ayuda a preparar tu juego multijugador para usarlo con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos y para obtener información

sobre el uso del complemento del SDK del servidor de C# para Unity, consulte Integración de Amazon GameLift en un proyecto de Unity.

Acciones

- GetSdkVersion()
- InitSDK()
- InitSDK()
- ProcessReady()
- ProcessEnding()
- ActivateGameSession()
- UpdatePlayerSessionCreationPolicy()
- GetGameSessionId()
- GetTerminationTime()
- AcceptPlayerSession()
- RemovePlayerSession()
- DescribePlayerSessions()
- StartMatchBackfill()
- StopMatchBackfill()
- GetComputeCertificate()
- GetFleetRoleCredentials()
- Destroy()

GetSdkVersion()

Devuelve el número de versión actual del SDK integrado en el proceso del servidor.

Sintaxis

```
AwsStringOutcome GetSdkVersion();
```

Valor devuelto

Si funciona correctamente, devuelve la versión del SDK actual como objeto the section called "AwsStringOutcome". La cadena devuelta solo incluye el número de versión (por ejemplo, `5.0.0`). Si no funciona, devuelve un mensaje de error.

Ejemplo

```
var getSdkVersionOutcome = GameLiftServerAPI.GetSdkVersion();
```

InitSDK()

Inicializa el Amazon GameLift SDK para una flota de EC2 gestionada. Llama a este método al lanzarlo, antes de que se GameLift produzca cualquier otra inicialización relacionada con Amazon. Este método lee los parámetros del servidor del entorno anfitrión para configurar la comunicación entre el servidor y el GameLift servicio de Amazon.

Sintaxis

```
GenericOutcome InitSDK();
```

Valor devuelto

Si tiene éxito, devuelve un InitSdkOutcome objeto que indica que el proceso del servidor está listo para la llamadaProcessReady().

Ejemplo

```
//Call InitSDK to establish a local connection with the GameLift agent to enable
 further communication.
GenericOutcome initSDKOutcome = GameLiftServerAPI.InitSDK();
```

InitSDK()

Inicializa el Amazon GameLift SDK para una Anywhere flota. Llama a este método al lanzarlo, antes de que se GameLift produzca cualquier otra inicialización relacionada con Amazon. Este método requiere parámetros de servidor explícitos para configurar la comunicación entre el servidor y el GameLift servicio de Amazon.

Sintaxis

```
GenericOutcome InitSDK(ServerParameters serverParameters);
```

Parámetros

ServerParameters

Para inicializar un servidor de juegos en una GameLift Anywhere flota de Amazon, construye un `ServerParameters` objeto con la siguiente información:

- La URL WebSocket utilizada para conectarte a tu servidor de juegos.
- El ID del proceso utilizado para alojar su servidor de juegos.
- El ID del proceso utilizado para alojar los procesos del servidor de juegos.
- El ID de la GameLift flota de Amazon que contiene tu ordenador de GameLift Anywhere Amazon.
- El token de autorización generado por la GameLift operación de Amazon.

Valor devuelto

Si se ejecuta correctamente, devuelve un InitSdkOutcome objeto que indica que el proceso del servidor está listo para la llamadaProcessReady().

> **Note**
>
> Si las llamadas a `InitSDK()` no funcionan en las compilaciones de juegos implementadas en las flotas de Anywhere, compruebe el parámetro `ServerSdkVersion` que se utiliza al crear el recurso de compilación. Debe establecer este valor de forma explícita en la versión del SDK del servidor en uso. El valor predeterminado de este parámetro es 4.x, que no es compatible. Para resolver este problema, cree una compilación nueva e impleméntela en una flota nueva.

Ejemplo

```
//Define the server parameters
string websocketUrl = "wss://us-west-1.api.amazongamelift.com";
string processId = "PID1234";
string fleetId = "aarn:aws:gamelift:us-west-1:111122223333:fleet/
fleet-9999ffff-88ee-77dd-66cc-5555bbbb44aa";
string hostId = "HardwareAnywhere";
string authToken = "1111aaaa-22bb-33cc-44dd-5555eeee66ff";
```

```
ServerParameters serverParameters =
  new ServerParameters(webSocketUrl, processId, hostId, fleetId, authToken);

//Call InitSDK to establish a local connection with the GameLift agent to enable
 further communication.
GenericOutcome initSDKOutcome = GameLiftServerAPI.InitSDK(serverParameters);
```

ProcessReady()

Notifica a Amazon de GameLift que el proceso del servidor está listo para albergar sesiones de juego. Llame a este método después de invocar InitSDK(). Se debe llamar a este método solo una vez por proceso.

Sintaxis

```
GenericOutcome ProcessReady(ProcessParameters processParameters)
```

Parámetros

ProcessParameters

Un objeto `ProcessParameters` contiene información sobre el proceso del servidor.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo muestra las implementaciones tanto de la función de método como de la función de delegación.

```
// Set parameters and call ProcessReady
ProcessParameters processParams = new ProcessParameters(
  this.OnStartGameSession,
  this.OnProcessTerminate,
  this.OnHealthCheck,
  this.OnUpdateGameSession,
  port,
  new LogParameters(new List<string>()
```

```
  // Examples of log and error files written by the game server
  {
    "C:\\game\\logs",
    "C:\\game\\error"
  })
);
GenericOutcome processReadyOutcome = GameLiftServerAPI.ProcessReady(processParams);
```

ProcessEnding()

Notifica a Amazon de GameLift que el proceso del servidor está finalizando. Utiliza este método después de realizar todas las demás tareas de limpieza (incluido el cierre de la sesión de juego activa) y antes de finalizar el proceso. Según el resultado de `ProcessEnding()`, el proceso finaliza con éxito (0) o error (-1) y genera un evento de flota. Si el proceso termina con un error, se generará el evento de flota. `SERVER_PROCESS_TERMINATED_UNHEALTHY`

Sintaxis

```
GenericOutcome ProcessEnding()
```

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

En este ejemplo, se llama a `ProcessEnding()` y `Destroy()` antes de finalizar el proceso del servidor con un código de salida correcto o erróneo.

```
GenericOutcome processEndingOutcome = GameLiftServerAPI.ProcessEnding();
GameLiftServerAPI.Destroy();

if (processEndingOutcome.Success)
  {
    Environment.Exit(0);
  }
else
  {
    Console.WriteLine("ProcessEnding() failed. Error: " +
 processEndingOutcome.Error.ToString());
    Environment.Exit(-1);
```

```
  }
```

ActivateGameSession()

Notifica a Amazon GameLift que el proceso del servidor ha activado una sesión de juego y ya está listo para recibir las conexiones de los jugadores. Esta acción debe llamarse como parte de la función de devolución de llamada `onStartGameSession()`, después de la inicialización de todas las sesiones de juego.

Sintaxis

```
GenericOutcome ActivateGameSession()
```

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo muestra cómo se llama a `ActivateGameSession()` como parte de la función de delegación `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()

Actualiza la capacidad de la sesión de juego actual para aceptar sesiones de jugador nuevas. Una sesión de juego se puede configurar para que acepte o deniegue todas las sesiones nuevas de los jugadores.

Sintaxis

```
GenericOutcome UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy
 playerSessionPolicy)
```

Parámetros

playerSessionPolicy

Valor de cadena que indica si la sesión de juego acepta jugadores nuevos.

Los valores válidos son:

- ACCEPT_ALL: se aceptan todas las sesiones de jugador nuevas.
- DENY_ALL: se rechazan todas las sesiones de jugador nuevas.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo establece la política de participación en la sesión de juego actual para aceptar todos los jugadores.

```
GenericOutcome updatePlayerSessionPolicyOutcome =

 GameLiftServerAPI.UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy.ACCEPT_ALL);
```

GetGameSessionId()

Recupera el ID de la sesión de juego alojada por el proceso del servidor.

En el caso de los procesos inactivos que no se activan con una sesión de juego, la llamada devuelve the section called “GameLiftError”.

Sintaxis

```
AwsStringOutcome GetGameSessionId()
```

Valor devuelto

Si funciona correctamente, devuelve el ID de sesión del juego como objeto the section called “AwsStringOutcome”. Si no funciona, devuelve un mensaje de error.

Ejemplo

```
AwsStringOutcome getGameSessionIdOutcome = GameLiftServerAPI.GetGameSessionId();
```

GetTerminationTime()

Devuelve la hora a la que está programada el cierre de un proceso de servidor, si hay una hora de terminación disponible. Un proceso de servidor realiza esta acción tras recibir una `onProcessTerminate()` llamada de Amazon GameLift. Amazon GameLift solicita `onProcessTerminate()` por los siguientes motivos:

- Cuando el proceso del servidor ha informado de un mal estado de salud o no ha respondido a Amazon GameLift.
- Al finalizar la instancia durante un evento de reducción vertical.
- Cuando se finaliza una instancia debido a la interrupción de una instancia de spot.

Sintaxis

```
AwsDateTimeOutcome GetTerminationTime()
```

Valor devuelto

Si el proceso se realiza correctamente, devuelve la hora de terminación como un objeto the section called “AwsDateTimeOutcome”. El valor es la hora de terminación expresado en ciclos transcurridos desde `0001 00:00:00`. Por ejemplo, el valor de la fecha y hora `2020-09-13 12:26:40 -000Z` es igual a `637355968000000000` ciclos. Si no hay una hora de terminación disponible, devuelve un mensaje de error.

Ejemplo

```
AwsDateTimeOutcome getTerminationTimeOutcome = GameLiftServerAPI.GetTerminationTime();
```

AcceptPlayerSession()

Notifica a Amazon GameLift que un jugador con el identificador de sesión de jugador especificado se ha conectado al proceso del servidor y necesita ser validado. Amazon GameLift verifica que el identificador de sesión del jugador sea válido. Una vez validada la sesión del jugador, Amazon GameLift cambia el estado del espacio del jugador de RESERVADO a ACTIVO.

Sintaxis

```
GenericOutcome AcceptPlayerSession(String playerSessionId)
```

Parámetros

playerSessionId

ID único que se emite GameLift cuando se crea una nueva sesión de jugador.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo muestra una función para gestionar una solicitud de conexión, incluida la validación y el rechazo de ID de sesión de jugador no válidos.

```
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()

Notifica a Amazon GameLift que un jugador se ha desconectado del proceso del servidor. En respuesta, Amazon GameLift cambia el espacio del jugador para que esté disponible.

Sintaxis

```
GenericOutcome RemovePlayerSession(String playerSessionId)
```

Parámetros

playerSessionId

ID único emitido por Amazon GameLift cuando se crea una nueva sesión de jugador.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
GenericOutcome removePlayerSessionOutcome =
 GameLiftServerAPI.RemovePlayerSession(playerSessionId);
```

DescribePlayerSessions()

Recupera datos de sesión de jugador, incluida la configuración, los metadatos de la sesión y los datos de jugador. Utilice esta acción para obtener información para una única sesión de jugador, para todas las sesiones de jugador de una sesión de juego o para todas las sesiones de jugador asociadas a un solo ID de jugador.

Sintaxis

```
DescribePlayerSessionsOutcome DescribePlayerSessions(DescribePlayerSessionsRequest
 describePlayerSessionsRequest)
```

Parámetros

DescribePlayerSessionsRequest

Un objeto the section called “DescribePlayerSessionsRequest” que describe las sesiones de jugador que recuperar.

Valor devuelto

Si funciona correctamente, devuelve un objeto the section called “DescribePlayerSessionsOutcome” que contiene un conjunto de objetos de sesión de jugador que se ajusta a los parámetros de la solicitud.

Ejemplo

Este ejemplo muestra una solicitud de todas las sesiones de jugador conectadas activamente a una sesión de juego específica. Al omitir NextTokeny establecer el valor límite en 10, Amazon GameLift devolverá los registros de las sesiones de los primeros 10 jugadores que coincidan con la solicitud.

```
// 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()

Envía una solicitud para encontrar nuevos jugadores para ranuras abiertas en una sesión de juego creada con FlexMatch. Para obtener más información, consulta la función FlexMatchde relleno.

Esta acción es asíncrona. Si se emparejan nuevos jugadores, Amazon GameLift proporciona datos actualizados de los emparejadores mediante la función de devolución de llamada. `OnUpdateGameSession()`

Un proceso del servidor solo puede tener una solicitud de reposición de emparejamiento activa a la vez. Para enviar una nueva solicitud, en primer lugar llame a StopMatchBackfill() para cancelar la solicitud original.

Sintaxis

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest
 startBackfillRequest);
```

Parámetros

StartMatchBackfillRequest

Un objeto `StartMatchBackfillRequest` contiene información sobre la solicitud de reposición.

Valor devuelto

Devuelve un objeto the section called “StartMatchBackfillOutcome” con el ID del ticket de reposición de emparejamiento o un error con un mensaje de error.

Ejemplo

```
// Build a backfill request
StartMatchBackfillRequest startBackfillRequest = new StartMatchBackfillRequest()
{
  TicketId = "1111aaaa-22bb-33cc-44dd-5555eeee66ff", //optional
  MatchmakingConfigurationArn = "arn:aws:gamelift:us-
west-2:111122223333:matchmakingconfiguration/MyMatchmakerConfig",
  GameSessionId = GameLiftServerAPI.GetGameSessionId().Result,    // gets ID for
 current game session
  MatchmakerData matchmakerData =
    MatchmakerData.FromJson(gameSession.MatchmakerData),  // gets matchmaker data for
 current players
  // get matchmakerData.Players
  // remove data for players who are no longer connected
  Players = ListOfPlayersRemainingInTheGame
};

// Send backfill request
StartMatchBackfillOutcome startBackfillOutcome =
 GameLiftServerAPI.StartMatchBackfill(startBackfillRequest);

// Implement callback function for backfill
void OnUpdateGameSession(GameSession myGameSession)
{
  // game-specific tasks to prepare for the newly matched players and update matchmaker
 data as needed
}
```

StopMatchBackfill()

Cancela una solicitud de reposición de emparejamiento activa. Para obtener más información, consulta la función de FlexMatch relleno.

Sintaxis

```
GenericOutcome StopMatchBackfill (StopMatchBackfillRequest stopBackfillRequest);
```

Parámetros

StopMatchBackfillRequest

Un objeto `StopMatchBackfillRequest` que proporciona detalles sobre el ticket de emparejamiento que va a detener.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
// Set backfill stop request parameters
StopMatchBackfillRequest stopBackfillRequest = new StopMatchBackfillRequest(){
  TicketId = "1111aaaa-22bb-33cc-44dd-5555eeee66ff", //optional, if not provided one is
 autogenerated
  MatchmakingConfigurationArn = "arn:aws:gamelift:us-
west-2:111122223333:matchmakingconfiguration/MyMatchmakerConfig",
  GameSessionId = GameLiftServerAPI.GetGameSessionId().Result    //gets the ID for the
 current game session
};
GenericOutcome stopBackfillOutcome =
 GameLiftServerAPI.StopMatchBackfillRequest(stopBackfillRequest);
```

GetComputeCertificate()

Recupera la ruta al certificado TLS utilizado para cifrar la conexión de red entre el servidor de juegos y el cliente de juego. Puedes usar la ruta del certificado al registrar tu dispositivo informático en una GameLift Anywhere flota de Amazon. Para obtener más información, consulte, RegisterCompute.

Sintaxis

```
GetComputeCertificateOutcome GetComputeCertificate();
```

Valor devuelto

Devuelve un GetComputeCertificateResponse objeto que contiene lo siguiente:

- CertificatePath: La ruta al certificado TLS de su recurso informático. Cuando se utiliza una flota GameLift gestionada por Amazon, esta ruta contiene:
  - `certificate.pem`: el certificado del usuario final. La cadena de certificados completa es la combinación del `certificateChain.pem` adjunto a este certificado.
  - `certificateChain.pem`: la cadena de certificados que contiene el certificado raíz y los certificados intermedios.
  - `rootCertificate.pem`: el certificado raíz.
  - `privateKey.pem`: la clave privada del certificado del usuario final.
- ComputeName: el nombre del recurso informático.

Ejemplo

```
GetComputeCertificateOutcome getComputeCertificateOutcome =
 GameLiftServerAPI.GetComputeCertificate();
```

GetFleetRoleCredentials()

Recupera las credenciales del rol de IAM que autorizan GameLift a Amazon a interactuar con otros. Servicios de AWS Para obtener más información, consulte Comunicación con otros recursos de AWS de sus flotas.

Sintaxis

```
GetFleetRoleCredentialsOutcome GetFleetRoleCredentials(GetFleetRoleCredentialsRequest
 request);
```

Parámetros

GetFleetRoleCredentialsRequest

Credenciales de rol que amplían el acceso limitado a sus recursos de AWS al servidor de juegos.

Valor devuelto

Devuelve un objeto the section called “GetFleetRoleCredentialsOutcome”.

Ejemplo

```
// form the fleet credentials request
GetFleetRoleCredentialsRequest getFleetRoleCredentialsRequest = new
 GetFleetRoleCredentialsRequest(){
  RoleArn = "arn:aws:iam::123456789012:role/service-role/exampleGameLiftAction"
};
GetFleetRoleCredentialsOutcome GetFleetRoleCredentialsOutcome credentials =
 GetFleetRoleCredentials(getFleetRoleCredentialsRequest);
```

Destroy()

Libera de memoria el SDK del servidor de GameLift juegos de Amazon. Como práctica recomendada, llame a este método después de `ProcessEnding()` y antes de finalizar el proceso. Si utilizas una flota de Anywhere y no vas a finalizar los procesos del servidor después de cada sesión de juego, llama `Destroy()` y, `InitSDK()` a continuación, reinicializa antes de notificar a Amazon GameLift que el proceso está listo para organizar una sesión de juego. `ProcessReady()`

Sintaxis

```
GenericOutcome Destroy()
```

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
// 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);
  }
```

Referencia del SDK del servidor de Amazon GameLift para C# y Unity: Tipos de datos

Esta referencia del SDK del servidor C# de Amazon GameLift como ayuda para preparar su juego multijugador para su uso con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos y para obtener información sobre el uso del complemento del SDK del servidor de C# para Unity, consulte Integración de Amazon GameLift en un proyecto de Unity.

Tipos de datos

- LogParameters
- ProcessParameters
- UpdateGameSession
- GameSession
- ServerParameters
- StartMatchBackfillRequest
- Jugador
- DescribePlayerSessionsRequest
- StopMatchBackfillRequest
- GetFleetRoleCredentialsRequest
- AttributeValue
- AwsStringOutcome
- GenericOutcome
- DescribePlayerSessionsOutcome
- DescribePlayerSessionsResult
- PlayerSession

- StartMatchBackfillOutcome
- StartMatchBackfillResult
- GetComputeCertificateOutcome
- GetComputeCertificateResult
- GetFleetRoleCredentialsOutcome
- GetFleetRoleCredentialsResult
- AwsDateTimeOutcome
- GameLiftError
- Enums

LogParameters

Utilice este tipo de datos para identificar los archivos generados durante una sesión de juegos que desee cargar en Amazon GameLift cuando finalice la sesión de juego. El servidor de juegos comunica `LogParameters to` Amazon GameLift en una llamada a ProcessReady().

Properties	Descripción
LogPaths	La lista de las rutas de directorio a archivos de registro del servidor de juegos que desea que Amazon GameLift almacene para futuros accesos. El proceso del servidor genera esos archivos durante una sesión de juego. Defina los nombres y las rutas de los archivos en el servidor de juegos y almacénelos en el directori o raíz de compilación del juego. Las rutas del registro deben ser absolutas. Por ejemplo, si la compilación del juego almacena los registros de sesión de juego en una ruta del tipo `MyGame\sessionLogs\` , la ruta sería `c:\game\MyGame\sessionLogs` en una instancia de Windows. Tipo: `List<String>`

	Obligatorio: no

ProcessParameters

Este tipo de datos contiene el conjunto de parámetros enviado a Amazon GameLift en una llamada a ProcessReady().

Properties	Descripción
LogParameters	El objeto con una lista de rutas de directorio a archivos de registro de la sesión de juego. Tipo: `Aws::GameLift::Server:: LogParameters` Obligatorio: sí
OnHealthCheck	El nombre de la función de devolución de llamada que invoca Amazon GameLift para solicitar un informe de estado del proceso de servidor. Amazon GameLift llama a esta función cada 60 segundos. Después de llamar a esta función, Amazon GameLift espera una respuesta durante 60 segundos y si no recibe ninguna, registra el proceso del servidor como en mal estado. Tipo: `void OnHealthCheckDelegate()` Obligatorio: sí
OnProcessTerminate	El nombre de la función de devolución de llamada que Amazon GameLift invoca para forzar el cierre del proceso de servidor. Después de llamar a esta función, Amazon GameLift espera cinco minutos hasta que el proceso de servidor se cierre y responde con

	una llamada a ProcessEnding() antes de cerrar el proceso de servidor. Tipo: `void OnProcessTerminate Delegate()` Obligatorio: sí
OnStartGameSession	El nombre de la función de devolución de llamada que Amazon GameLift invoca para activar una nueva sesión de juego. Amazon GameLift llama a esta función como respuesta a la solicitud de cliente CreateGameSession. La función de devolución de llamada pasa un objeto GameSession (definido en la Referencia de la API de Amazon GameLift). Tipo: `void OnStartGameSession Delegate( GameSession )` Obligatorio: sí
OnUpdateGameSession	El nombre de la función de devolución de llamada que Amazon GameLift invoca para pasar un objeto de sesión de juego actualizado al proceso de servidor. Amazon GameLift llama a esta función cuando se ha procesado una solicitud de reposición de emparejamiento para proporcionar datos actualizados de los emparejadores. Pasa un objeto GameSession, una actualización de estado (`updateReason`) y el ID del ticket de reposición de emparejamiento. Tipo: void OnUpdateGameSessionDelegate (UpdateGameSession) Obligatorio: no

Port	El número de puerto en el que escucha el proceso del servidor para recibir conexiones de jugador nuevas. El valor debe estar en el rango de puertos configurado para cualquier flota que implemente esa compilación de servidor de juegos. Este número de puerto se incluye en los objetos de sesión de juego y de jugador, que las sesiones de juego utilizan a la hora de conectarse a un proceso del servidor. Tipo: `Integer` Obligatorio: sí

UpdateGameSession

La información actualizada de un objeto de sesión de juego incluido el motivo por el que se actualizó la sesión de juego. Si la actualización está relacionada con una acción de reposición del emparejamiento, este tipo de datos incluye el ID de ticket de reposición.

Propiedades	Descripción
GameSession	Un objeto GameSession definido por la API de Amazon GameLift. El objeto `GameSessi on` contiene propiedades que describen una sesión de juego. Tipo: `GameSession GameSession()` Obligatorio: sí
UpdateReason	El motivo por el que se actualiza la sesión de juego. Tipo: `UpdateReason UpdateReason()` Obligatorio: sí

Propiedades	Descripción
BackfillTicketId	El ID de ticket de reposición que intenta actualizar la sesión de juego. Tipo: `String` Obligatorio: sí

GameSession

Detalles de una sesión de juego.

Propiedades	Descripción
GameSessionId	Un identificador único de la sesión de juego. El ARN de una sesión de juego tiene el siguiente formato: `arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token>` . Tipo: `String` Obligatorio: no
Nombre	Una etiqueta descriptiva de la sesión de juego. Tipo: `String` Obligatorio: no
FleetId	Un identificador único para la flota en la que se ejecuta la sesión de juego. Tipo: `String` Obligatorio: no

Propiedades	Descripción
MaximumPlayerSessionCount	El número máximo de conexiones de jugadores a la sesión de juego. Tipo: `Integer` Obligatorio: no
Port	El número de puerto de la sesión de juego. Para conectarse a un servidor de juegos de Amazon GameLift, una aplicación necesita tanto la dirección IP como el número de puerto. Tipo: `Integer` Obligatorio: no
IpAddress	La dirección IP del servidor de la sesión de juego. Para conectarse a un servidor de juegos de Amazon GameLift, una aplicación necesita tanto la dirección IP como el número de puerto. Tipo: `String` Obligatorio: no
GameSessionData	Un conjunto de propiedades de sesión de juego personalizadas, formateadas como un valor de una sola cadena. Tipo: `String` Obligatorio: no

Propiedades	Descripción
MatchmakerData	La información sobre el proceso de emparejamiento que se utilizó para crear la sesión de juego, en sintaxis JSON, con formato como cadena. Además de la configuración de emparejamiento utilizada, contiene datos sobre todos los jugadores asignados al emparejamiento, incluidos los atributos de los jugadores y las asignaciones de los equipos. Tipo: `String` Obligatorio: no
GameProperties	Un conjunto de propiedades personalizadas de una sesión de juego, con formato como pares clave-valor. Estas propiedades se pasan a una solicitud de iniciar una nueva sesión de juego. Tipo: `Dictionary<string, string>` Obligatorio: no

Propiedades	Descripción
DnsName	El identificador de DNS asignado a la instancia que ejecuta la sesión de juego. Los valores tienen formato siguiente: • Flotas habilitadas para TLS: `<unique identifier>.<region identifier>.amazongamelift.com` . • Flotas no habilitadas para TLS: `ec2-<unique identifier>.compute.amazonaws.com` . Cuando se conecte a una sesión de juego que se ejecute en una flota habilitada de TLS, debe utilizar el nombre de DNS, no la dirección IP. Tipo: `String` Obligatorio: no

ServerParameters

La información utilizada para mantener la conexión entre el servidor de Amazon GameLift Anywhere y el servicio de Amazon GameLift. Esta información se utiliza al inicializar nuevos procesos de servidor con InitSDK(). Para los servidores alojados en instancias de EC2 administradas por Amazon GameLift, utilice un objeto vacío.

Propiedades	Descripción
WebSocketUrl	Se devuelve el `GameLiftServerSdkEndpoint` cuando `RegisterCompute` como parte de Amazon GameLift Anywhere. Tipo: `String` Obligatorio: sí

Propiedades	Descripción
ProcessId	Un identificador único registrado en el proceso de servidor que aloja el juego. Tipo: `String` Obligatorio: sí
HostId	Un identificador único para el alojamiento con el servidor que procesa el juego. El hostId es el ComputeName utilizado cuando registró el recurso informático. Para obtener más información, consulte, RegisterCompute Tipo: `String` Obligatorio: sí
FleetId	El ID de la flota de la flota en la que está registrado el recurso informático. Para obtener más información, consulte RegisterCompute. Tipo: `String` Obligatorio: sí
AuthToken	El token de autenticación generado por Amazon GameLift que autentica el servidor en Amazon GameLift. Para obtener más informaci ón, consulte GetComputeAuthToken. Tipo: `String` Obligatorio: sí

StartMatchBackfillRequest

Información utilizada para crear una solicitud de reposición de emparejamiento. El servidor de juegos comunica esa información a Amazon GameLift en una llamada a StartMatchBackfill().

Propiedades	Descripción
GameSessionArn	El identificador único de la sesión de juego. El `GetGameSessionId` de la operación de la API devuelve el identificador en formato de ARN. Tipo: `String` Obligatorio: sí
MatchmakingConfigurationArn	El identificador único, con formato de ARN, que el emparejador utiliza para esta solicitud. El ARN del emparejador para la sesión de juego original se encuentra en el objeto de sesión de juego en la propiedad de datos del emparejador. Puede obtener más información sobre los datos del emparejador en Trabajo con datos del emparejador. Tipo: `String` Obligatorio: sí
Players	Un conjunto de datos que representa a todos los jugadores que están actualmente en la sesión de juego. El creador de emparejamientos utiliza esta información para buscar nuevos jugadores que son idóneos para los jugadores actuales. Tipo: `List<Player>` Obligatorio: sí
TicketId	El identificador único para un ticket de solicitud de emparejamiento o reposición de emparejamiento. Si no proporciona un valor, Amazon GameLift generará uno. Use este identific

Propiedades	Descripción
	ador para realizar un seguimiento del estado del ticket de reposición de emparejamiento o cancelar la solicitud si es necesario. Tipo: `String` Obligatorio: no

Jugador

Representa a un jugador en el emparejamiento. Cuando se inicia una solicitud de emparejamiento, un jugador tiene un ID de jugador, atributos y, posiblemente, datos de latencia. Amazon GameLift añade la información del equipo después de que se realice un emparejamiento.

Propiedades	Descripción
LatencyInMS	Un conjunto de valores expresados en milisegundos que indican la cantidad de latencia que experimenta un jugador cuando se conecta a una ubicación. Si se utiliza esta propiedad, el jugador solo se empareja con las ubicaciones que aparecen. Si un creador de emparejamientos tiene una regla que evalúa la latencia de los jugadores , estos deben informar de la latencia para ser emparejados. Tipo: `Dictionary<string, int>` Obligatorio: no
PlayerAttributes	Una colección de pares de clave-valor que contienen información del jugador para su uso en el emparejamiento. Las claves de atributos de los jugadores deben emparejarse con los

Propiedades	Descripción
	atributos PlayerAttributes utilizados en un conjunto de reglas de emparejamiento. Para obtener más información sobre los atributos del jugador, consulte AttributeValue. Tipo: `Dictionary<string, AttributeValue` Obligatorio: no
PlayerId	Un identificador único de un jugador. Tipo: `String` Obligatorio: no
Equipo	El nombre del equipo al que está asignado el jugador en un emparejamiento. Defina el nombre del equipo se define en el conjunto de reglas de emparejamiento. Tipo: `String` Obligatorio: no

DescribePlayerSessionsRequest

Este tipo de datos se utiliza para especificar qué sesión o sesiones de jugador recuperar. Se puede utilizar de varias maneras: (1) proporcione un PlayerSessionId para solicitar una sesión de jugador específica; (2) proporcione un GameSessionId para solicitar todas las sesiones de jugador de la sesión de juego especificada; o (3) proporcione un PlayerId para solicitar todas las sesiones de jugador del jugador especificado. Para grandes recopilaciones de sesiones de jugador, utilice los parámetros de paginación para recuperar resultados como páginas secuenciales.

Propiedades	Descripción
GameSessionId	El identificador único de la sesión de juego. Use este parámetro para solicitar todas las sesiones de jugador de la sesión de juego especificada. El formato de ID de la sesión de juego es el siguiente: `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>` . El valor de la **<**ID string**>** es una cadena de ID personalizada (si se especificó una cuando se creó la sesión de juego) o una cadena generada. Tipo: `String` Obligatorio: no
PlayerSessionId	El identificador único de una sesión de jugador. Tipo: `String` Obligatorio: no
PlayerId	El identificador único de un jugador. Consulte Generación de ID de jugador. Tipo: `String` Obligatorio: no
PlayerSessionStatusFilter	El estado de la sesión de jugador para filtrar los resultados. Los posibles estados de sesión de jugador son: • RESERVED: se ha recibido la solicitud de sesión de jugador, pero el jugador aún no se ha conectado al proceso del servidor o aún no se ha validado.

Propiedades	Descripción
	• ACTIVE: el proceso del servidor ha validado el jugador y actualmente está conectado. • COMPLETED: ha caído la conexión del jugador. • TIMEDOUT: se ha recibido una solicitud de sesión de jugador, pero el jugador no se ha conectado y/o no se ha validado en el plazo de tiempo de espera (60 segundos). Tipo: `String` Obligatorio: no
NextToken	El token que indica el inicio de la siguiente página de resultados. Para especificar el inicio del conjunto de resultados, no indique ningún valor. Si se especifica un ID de sesión de jugador, este parámetro se ignora. Tipo: `String` Obligatorio: no
Límite	El número máximo de resultados que devolver. Si se especifica un ID de sesión de jugador, este parámetro se ignora. Tipo: `int` Obligatorio: no

StopMatchBackfillRequest

Información utilizada para cancelar una solicitud de reposición de emparejamiento. El servidor de juegos comunica esa información al servicio de Amazon GameLift en una llamada a StopMatchBackfill().

Propiedades	Descripción
GameSessionArn	El identificador único de sesión de juego de la solicitud que se va a cancelar. Tipo: `string` Obligatorio: sí
MatchmakingConfigurationArn	El identificador único del emparejador al que se envió esta solicitud. Tipo: `string` Obligatorio: sí
TicketId	El identificador único del ticket de solicitud de reposición que se va a cancelar. Tipo: `string` Obligatorio: sí

GetFleetRoleCredentialsRequest

Este tipo de datos proporciona al servidor de juegos un acceso limitado a los otros recursos de AWS. Para obtener más información, consulte, Configurar un rol de servicio de IAM para Amazon GameLift.

Propiedades	Descripción
RoleArn	El nombre de recurso de Amazon (ARN) del rol de servicio que amplía el acceso limitado a sus recursos de AWS. Tipo: `string` Obligatorio: sí
RoleSessionName	El nombre de la sesión que describe el uso de las credenciales del rol.

Propiedades	Descripción
	Tipo: `string` Obligatorio: no

AttributeValue

Utilice estos valores en pares de clave-valor de atributo Jugador. Este objeto le permite especificar un valor de atributo mediante cualquiera de los tipos de datos válidos: cadena, número, matriz de cadenas o mapa de datos. Cada objeto `AttributeValue` puede utilizar solo una de las propiedades disponibles.

Propiedades	Descripción
attrType	Especifica el tipo de valor del atributo. Tipo: un valor de enum de `AttrType`. Obligatorio: no
S	Representa un valor de atributo de cadena. Tipo: `string` Obligatorio: sí
N	Representa un valor de atributo numérico. Tipo: `double` Obligatorio: sí
SL	Representa una matriz de valores de atributos de cadena. Tipo: `string[]` Obligatorio: sí

Propiedades	Descripción
SDM	Representa un diccionario de claves de cadena y valores dobles. Tipo: `Dictionary<string, double>` Obligatorio: sí

AwsStringOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Resultado	El resultado de la acción. Tipo: `string` Obligatorio: no
Correcto	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

GenericOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Correcto	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

DescribePlayerSessionsOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Resultado	El resultado de la acción. Tipo: the section called “DescribePlayerSessionsResult” Obligatorio: no
Correcto	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

DescribePlayerSessionsResult

Propiedades	Descripción
NextToken	El token que indica el inicio de la siguiente página de resultados. Para especificar el inicio del conjunto de resultados, no indique ningún valor. Si se especifica un ID de sesión de jugador, este parámetro se ignora. Tipo: `string` Obligatorio: sí
PlayerSessions	Una colección de objetos que contiene propiedades para cada sesión de jugador que se empareja con la solicitud. Tipo: `IList<`the section called “PlayerSession” `>` Obligatorio:
Correcto	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

PlayerSession

Propiedades	Descripción
CreationTime	Tipo: `long` Obligatorio: sí
FleetId	Tipo: `string` Obligatorio: sí
GameSessionId	Tipo: `string` Obligatorio: sí
IpAddress	Tipo: `string` Obligatorio: sí
Datos del jugador	Tipo: `string` Obligatorio: sí
PlayerId	Tipo: `string` Obligatorio: sí
PlayerSessionId	Tipo: `string` Obligatorio: sí
Port	Tipo: `int` Obligatorio: sí
Estado	Tipo: una enumeración de `PlayerSessionStatus` . Obligatorio: sí
TerminationTime	Tipo: `long`

Propiedades	Descripción
	Obligatorio: sí
DnsName	Tipo: `string` Obligatorio: sí

StartMatchBackfillOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Resultado	El resultado de la acción. Tipo: the section called “StartMatchBackfillResult” Obligatorio: no
Correcto	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

StartMatchBackfillResult

Propiedades	Descripción
TicketId	Tipo: `string`

Propiedades	Descripción
	Obligatorio: sí

GetComputeCertificateOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Resultado	El resultado de la acción. Tipo: the section called “GetComputeCertificateResult” Obligatorio: no
Correcto	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

GetComputeCertificateResult

La ruta al certificado TLS de su recurso informático y el nombre de host del equipo.

Propiedades	Descripción
CertificatePath	Tipo: `string` Obligatorio: sí

Propiedades	Descripción
ComputeName	Tipo: `string` Obligatorio: sí

GetFleetRoleCredentialsOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Resultado	El resultado de la acción. Tipo: the section called “GetFleetRoleCredentialsResult” Obligatorio: no
Correcto	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

GetFleetRoleCredentialsResult

Propiedades	Descripción
AccessKeyId	El ID de la clave de acceso para autenticar y proporcionar acceso a los recursos de AWS.

Propiedades	Descripción
	Tipo: `string` Obligatorio: no
AssumedRoleId	El ID del usuario al que pertenece el rol de servicio. Tipo: `string` Obligatorio: no
AssumedRoleUserArn	El nombre de recurso de Amazon (ARN) del usuario al que pertenece el rol de servicio. Tipo: `string` Obligatorio: no
Expiration	El tiempo que queda hasta que caduquen las credenciales de la sesión. Tipo: `DateTime` Obligatorio: no
SecretAccessKey	El ID de clave de acceso secreta para la autenticación. Tipo: `string` Obligatorio: no
SessionToken	Un token para identificar la sesión activa actual que interactúa con los recursos de AWS. Tipo: `string` Obligatorio: no

Propiedades	Descripción
Correcto	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

AwsDateTimeOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Resultado	El resultado de la acción. Tipo: `DateTime` Obligatorio: no
Correcto	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

GameLiftError

Propiedades	Descripción
ErrorType	Tipo de error. Tipo: una enumeración de `GameLiftErrorType` . Obligatorio: no
ErrorName	El nombre del error. Tipo: `string` Obligatorio: no
ErrorMessage	Mensaje de error. Tipo: `string` Obligatorio: no

Enums

Las enumeraciones definidas para el SDK del servidor de Amazon GameLift (C#) se definen de la siguiente manera:

AttrType

- NONE
- STRING
- DOUBLE
- STRING_LIST
- STRING_DOUBLE_MAP

GameLiftErrorType

Valor de cadena que indica el tipo de error. Los valores válidos son:

- SERVICE_CALL_FAILED: se ha producido un error en la llamada a un servicio de AWS.

- LOCAL_CONNECTION_FAILED: se ha producido un error en la conexión local a Amazon GameLift.
- NETWORK_NOT_INITIALIZED: la red no se ha inicializado.
- GAMESESSION_ID_NOT_SET: no se ha establecido el ID de sesión de juego.
- BAD_REQUEST_EXCEPTION
- INTERNAL_SERVICE_EXCEPTION
- ALREADY_INITIALIZED: el servidor o cliente de Amazon GameLift ya se inicializó con Inicializar().
- FLEET_MISMATCH: la flota de destino no coincide con la flota de gameSession o playerSession.
- GAMELIFT_CLIENT_NOT_INITIALIZED: el cliente de Amazon GameLift no se ha inicializado.
- GAMELIFT_SERVER_NOT_INITIALIZED: el servidor de Amazon GameLift no se ha inicializado.
- GAME_SESSION_ENDED_FAILED: el SDK del servidor de Amazon GameLift no pudo ponerse en contacto con el servicio para informar de que la sesión de juego había finalizado.
- GAME_SESSION_NOT_READY: no se activó la sesión de juego del servidor de Amazon GameLift.
- GAME_SESSION_READY_FAILED: el SDK del servidor de Amazon GameLift no pudo ponerse en contacto con el servicio para informar de que la sesión de juego estaba lista.
- INITIALIZATION_MISMATCH: se llamó a un método de cliente después de Server::Initialize() o viceversa.
- NOT_INITIALIZED: el servidor o cliente de Amazon GameLift no se ha iniciado con Inicializar().
- NO_TARGET_ALIASID_SET: no se ha establecido un aliasId de destino.
- NO_TARGET_FLEET_SET: no se ha establecido una flota de destino.
- PROCESS_ENDING_FAILED: el SDK del servidor de Amazon GameLift no pudo contactar con el servicio para informar de que el proceso está finalizando.
- PROCESS_NOT_ACTIVE: el proceso de servidor aún no está activo, no está vinculado a una GameSession y no puede aceptar ni procesar PlayerSessions.
- PROCESS_NOT_READY: el proceso de servidor aún no está listo para activarse.
- PROCESS_READY_FAILED: el SDK del servidor de Amazon GameLift no pudo contactar con el servicio para informar de que el proceso está listo.
- SDK_VERSION_DETECTION_FAILED: no se pudo detectar la versión del SDK.

- STX_CALL_FAILED: no se pudo realizar una llamada al componente de backend del servidor xSTx.
- STX_INITIALIZATION_FAILED: no se pudo inicializar el componente de backend del servidor xSTx.
- UNEXPECTED_PLAYER_SESSION: el servidor ha detectado una sesión de jugador no registrada.
- WEBSOCKET_CONNECT_FAILURE
- WEBSOCKET_CONNECT_FAILURE_FORBIDDEN
- WEBSOCKET_CONNECT_FAILURE_INVALID_URL
- WEBSOCKET_CONNECT_FAILURE_TIMEOUT
- WEBSOCKET_RETRIABLE_SEND_MESSAGE_FAILURE: error al enviar un mensaje al WebSocket del servicio de GameLift.
- WEBSOCKET_SEND_MESSAGE_FAILURE: error al enviar un mensaje al WebSocket del servicio de GameLift.
- MATCH_BACKFILL_REQUEST_VALIDATION: se ha producido un error en la validación de la solicitud.
- PLAYER_SESSION_REQUEST_VALIDATION: se ha producido un error en la validación de la solicitud.

PlayerSessionCreationPolicy

Valor de cadena que indica si la sesión de juego acepta jugadores nuevos. Los valores válidos son:

- ACCEPT_ALL: se aceptan todas las sesiones de jugador nuevas.
- DENY_ALL: se rechazan todas las sesiones de jugador nuevas.
- NOT_SET: la sesión de juego no está configurada para aceptar o denegar sesiones de nuevos jugadores.

PlayerSessionStatus

- ACTIVE
- COMPLETED
- NOT_SET
- RESERVED
- TIMEDOUT

Referencia del SDK del servidor Amazon GameLift 4.x para C#

Esta referencia del SDK del servidor C# de Amazon GameLift 4.x como ayuda para preparar su juego multijugador para su uso con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

Temas

- Referencia del SDK del servidor de Amazon GameLift (C#): Acciones
- Referencia del SDK del servidor de Amazon GameLift (C#): Tipos de datos

Referencia del SDK del servidor de Amazon GameLift (C#): Acciones

Puede utilizar esta referencia del SDK del servidor C# de Amazon GameLift como ayuda para preparar su juego multijugador para su uso con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

- Acciones
- Tipos de datos

AcceptPlayerSession()

Notifica al servicio de Amazon GameLift que un jugador con el ID de sesión de jugador especificado se ha conectado al proceso del servidor y requiere validación. Amazon GameLift verifica que el ID de sesión del jugador es válido, es decir, que el ID de jugador ha reservado una ranura de jugador en la sesión de juego. Una vez completada la validación, Amazon GameLift cambia el estado de la ranura de jugador de RESERVADO a ACTIVO.

Sintaxis

```
GenericOutcome AcceptPlayerSession(String playerSessionId)
```

Parámetros

playerSessionId

ID único emitido por Amazon GameLift cuando se crea una nueva sesión de jugador. El ID de sesión de un jugador se especifica en un objeto `PlayerSession`, que se

devuelve como respuesta a una llamada de cliente a las acciones de la API de GameLift StartGameSessionPlacement, CreateGameSession, DescribeGameSessionPlacement o DescribePlayerSessions.

Tipo: String

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo muestra una función para gestionar una solicitud de conexión, incluida la validación y el rechazo de ID de sesión de jugador no válidos.

```
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()

Informa al servicio de Amazon GameLift de que el proceso del servidor ha activado una sesión de juego y que está listo para recibir las conexiones de los jugadores. Esta acción debe llamarse como parte de la función de devolución de llamada `onStartGameSession()`, después de completar la inicialización de todas las sesiones de juego.

Sintaxis

```
GenericOutcome ActivateGameSession()
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo muestra cómo se llama a `ActivateGameSession()` como parte de la función de delegación `onStartGameSession()`.

```
void OnStartGameSession(GameSession gameSession)
{
    // game-specific tasks when starting a new game session, such as loading map

    // When ready to receive players
    var activateGameSessionOutcome = GameLiftServerAPI.ActivateGameSession();
}
```

DescribePlayerSessions()

Recupera datos de sesión de jugador, incluida la configuración, los metadatos de la sesión y los datos de jugador. Utilice esta acción para obtener información para una única sesión de jugador, para todas las sesiones de jugador de una sesión de juego o para todas las sesiones de jugador asociadas a un solo ID de jugador.

Sintaxis

```
DescribePlayerSessionsOutcome DescribePlayerSessions(DescribePlayerSessionsRequest
 describePlayerSessionsRequest)
```

Parámetros

describePlayerSessionsRequest

Es un objeto DescribePlayerSessionsRequest que describe las sesiones de jugador a recuperar.

Obligatorio: sí

Valor devuelto

Si funciona correctamente, devuelve un objeto `DescribePlayerSessionsOutcome` que contiene un conjunto de objetos de sesión de jugador que se ajusta a los parámetros de la solicitud. Los objetos de las sesiones de jugador tienen una estructura idéntica al tipo de datos PlayerSession de la API de Amazon GameLift del SDK de AWS.

Ejemplo

Este ejemplo muestra una solicitud de todas las sesiones de jugador conectadas activamente a una sesión de juego específica. Al omitir NextToken y definir el valor Límite en 10, Amazon GameLift devolverá los primeros 10 registros de sesiones de jugador que coincidan con la solicitud.

```
// Set request parameters
var describePlayerSessionsRequest = new
 Aws.GameLift.Server.Model.DescribePlayerSessionsRequest()
{
    GameSessionId = GameLiftServerAPI.GetGameSessionId().Result,    //gets the ID for
 the current game session
    Limit = 10,
    PlayerSessionStatusFilter =
 PlayerSessionStatusMapper.GetNameForPlayerSessionStatus(PlayerSessionStatus.ACTIVE)
};
// Call DescribePlayerSessions
Aws::GameLift::DescribePlayerSessionsOutcome playerSessionsOutcome =
    Aws::GameLift::Server::Model::DescribePlayerSessions(describePlayerSessionRequest);
```

GetGameSessionId()

Recupera el ID de la sesión de juego alojada actualmente por el proceso del servidor, siempre que esté activo.

En el caso de los procesos inactivos que no activados aún con una sesión de juego, la llamada devuelve `Success=True` y `GameSessionId=""` (una cadena vacía).

Sintaxis

```
AwsStringOutcome GetGameSessionId()
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Si funciona correctamente, devuelve el ID de sesión del juego como objeto `AwsStringOutcome`. Si no funciona, devuelve un mensaje de error.

Ejemplo

```
var getGameSessionIdOutcome = GameLiftServerAPI.GetGameSessionId();
```

GetInstanceCertificate()

Recupera la ubicación del archivo de un certificado TLS codificado con PEM que está asociado a la flota y sus instancias. AWS Certificate Manager genera este certificado al crear una nueva flota con la configuración del certificado establecida en GENERADO. Utilice este certificado para establecer una conexión segura con un cliente de juego y para cifrar la comunicación cliente/servidor.

Sintaxis

```
GetInstanceCertificateOutcome GetInstanceCertificate();
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Si se ejecuta correctamente, devuelve un objeto `GetInstanceCertificateOutcome` que contiene la ubicación del archivo de certificado TLS y la cadena de certificados de la flota, que se almacena en la instancia. En la instancia también se almacena un archivo de certificado raíz extraído de la cadena de certificados. Si no funciona, devuelve un mensaje de error.

Para obtener más información sobre el certificado y los datos de la cadena de certificados, consulte Elementos de respuesta de GetCertificate en la referencia de la API de AWS Certificate Manager.

Ejemplo

```
var getInstanceCertificateOutcome = GameLiftServerAPI.GetInstanceCertificate();
```

GetSdkVersion()

Devuelve el número de versión actual del SDK integrado en el proceso del servidor.

Sintaxis

```
AwsStringOutcome GetSdkVersion()
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Si funciona correctamente, devuelve la versión del SDK actual como objeto `AwsStringOutcome`. La cadena devuelta solo incluye el número de versión (por ejemplo, («3.1.5»). Si no funciona, devuelve un mensaje de error.

Ejemplo

```
var getSdkVersionOutcome = GameLiftServerAPI.GetSdkVersion();
```

GetTerminationTime()

Devuelve la hora a la que está programada el cierre de un proceso de servidor, si hay una hora de terminación disponible. Un proceso de servidor realiza esta acción después de recibir una devolución de llamada on`ProcessTerminate()` desde el servicio de Amazon GameLift. Amazon GameLift puede llamar a on`ProcessTerminate()` por los siguientes motivos: (1) estado deficiente (el proceso del servidor ha informado de un mal estado o no ha respondido a Amazon GameLift, (2) cuando se termina la instancia durante un evento de reducción vertical o (3) cuando una instancia se cierra debido a una interrupción de instancia de spot.

Si el proceso ha recibido una devolución de llamada on`ProcessTerminate()`, el devuelto es la hora de terminación estimada. Si el proceso no ha recibido ninguna devolución de llamada on`ProcessTerminate()`, se devuelve un mensaje de error. Más información acerca del apagado de un proceso de servidor.

Sintaxis

```
AwsDateTimeOutcome GetTerminationTime()
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Si el proceso se realiza correctamente, devuelve la hora de terminación como un objeto `AwsDateTimeOutcome`. El valor es la hora de terminación expresado en ciclos transcurridos desde 0001 00:00:00. Por ejemplo, el valor de fecha y hora 13-09-2020 12:26:40 -000Z es igual a 637355968000000000 ciclos. Si no hay una hora de terminación disponible, devuelve un mensaje de error.

Ejemplo

```
var getTerminationTimeOutcome = GameLiftServerAPI.GetTerminationTime();
```

InitSDK()

Inicializa el SDK de Amazon GameLift. Este método debe llamarse en el momento del lanzamiento, antes de cualquier otra inicialización relacionada con Amazon GameLift.

Sintaxis

```
InitSDKOutcome InitSDK()
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Si funciona correctamente, devuelve un objeto InitSdkOutcome e indica que el proceso del servidor está listo para llamar a ProcessReady().

Ejemplo

```
var initSDKOutcome = GameLiftServerAPI.InitSDK();
```

ProcessEnding()

Informa al servicio de Amazon GameLift de que el proceso del servidor se va a detener. Este método debe llamarse después de realizar las demás tareas de limpieza, incluido el cierre de todas las sesiones de juego activas. Se debe salir de este método con un código de salida de 0; un código

de salida que no sea 0 provoca un mensaje de evento que afirma que no se ha salido del proceso correctamente.

Después de que el método finalice con un código de 0, puede terminar el proceso con un código de salida correcto. También puede salir del proceso con un código de error. Si sale con un código de error, el evento de la flota indicará que el proceso ha finalizado incorrectamente (`SERVER_PROCESS_TERMINATED_UNHEALTHY`).

Sintaxis

```
GenericOutcome ProcessEnding()
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

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

ProcessReady()

Notifica al servicio de Amazon GameLift que el proceso del servidor está listo para alojar sesiones de juego. Llame a este método después de invocar correctamente a InitSDK() y completar las tareas de configuración necesarias antes de que el proceso del servidor pueda alojar una sesión de juego. Se debe llamar a este método solo una vez por proceso.

Sintaxis

```
GenericOutcome ProcessReady(ProcessParameters processParameters)
```

Parámetros

processParameters

Es un objeto ProcessParameters que comunica la siguiente información acerca del proceso del servidor:

- Nombres de métodos de devolución de llamada, implementados en el código de servidor de juegos, que el servicio de Amazon GameLift invoca para comunicarse con el proceso del servidor.
- Número de puerto de escucha del servidor de proceso.
- Ruta a cualquier archivo específico de la sesión de juego que desea que Amazon GameLift capture y almacene.

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo ilustra las implementaciones tanto de la función de llamada ProcessReady() como de la función de delegación.

```
// 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()

Informa al servicio de Amazon GameLift de que un jugador con el ID de sesión de jugador especificado se ha desconectado del proceso del servidor. Como respuesta, Amazon GameLift cambia el estado de la ranura de jugador a disponible, por lo que se le puede asignar un jugador nuevo.

Sintaxis

```
GenericOutcome RemovePlayerSession(String playerSessionId)
```

Parámetros

playerSessionId

ID único emitido por Amazon GameLift cuando se crea una nueva sesión de jugador. El ID de sesión de un jugador se especifica en un objeto `PlayerSession`, que se devuelve como respuesta a una llamada de cliente a las acciones de la API de GameLift StartGameSessionPlacement, CreateGameSession, DescribeGameSessionPlacement o DescribePlayerSessions.

Tipo: String

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
Aws::GameLift::GenericOutcome disconnectOutcome =
    Aws::GameLift::Server::RemovePlayerSession(playerSessionId);
```

StartMatchBackfill()

Envía una solicitud para encontrar nuevos jugadores para ranuras abiertas en una sesión de juego creada con FlexMatch. Consulte también la acción del SDK de AWS StartMatchBackfill(). Con esta acción, un proceso del servidor de juegos que aloja la sesión de juego puede iniciar solicitudes de reposición de emparejamiento. Puede obtener más información sobre la característica de reposición de FlexMatch.

Esta acción es asíncrona. Si se emparejan correctamente nuevos jugadores, el servicio de Amazon GameLift ofrece datos actualizados del emparejador por medio de la función de devolución de llamada `OnUpdateGameSession()`.

Un proceso del servidor solo puede tener una solicitud de reposición de emparejamiento activa a la vez. Para enviar una nueva solicitud, en primer lugar llame a StopMatchBackfill() para cancelar la solicitud original.

Sintaxis

```
StartMatchBackfillOutcome StartMatchBackfill (StartMatchBackfillRequest
 startBackfillRequest);
```

Parámetros

StartMatchBackfillRequest

Un objeto StartMatchBackfillRequest que comunica la siguiente información:

- Un ID de ticket que se asignará a la solicitud de reposición. Esta información es opcional; si no se proporciona ningún ID, Amazon GameLift generará automáticamente uno.
- El creador de emparejamientos al que se enviará la solicitud. El ARN de configuración completo es obligatorio. Este valor se puede obtener de los datos del creador de emparejamientos de la sesión de juego.
- El ID de la sesión de juego que está en fase de reposición.
- Datos de emparejamiento disponibles para los jugadores actuales de la sesión de juego.

Obligatorio: sí

Valor devuelto

Devuelve un objeto StartMatchBackfillOutcome con el ID del ticket de reposición de emparejamiento o un error con un mensaje de error.

Ejemplo

```
// Build a backfill request
var startBackfillRequest = new AWS.GameLift.Server.Model.StartMatchBackfillRequest()
{
    TicketId = "a ticket ID", //optional
    MatchmakingConfigurationArn = "the matchmaker configuration ARN",
    GameSessionId = GameLiftServerAPI.GetGameSessionId().Result,    // gets ID for
 current game session
        //get player data for all currently connected players
            MatchmakerData matchmakerData =
              MatchmakerData.FromJson(gameSession.MatchmakerData);  // gets matchmaker
 data for current players
            // get matchmakerData.Players
            // remove data for players who are no longer connected
    Players = ListOfPlayersRemainingInTheGame
};

// Send backfill request
var startBackfillOutcome = GameLiftServerAPI.StartMatchBackfill(startBackfillRequest);

// Implement callback function for backfill
void OnUpdateGameSession(GameSession myGameSession)
{
   // game-specific tasks to prepare for the newly matched players and update
 matchmaker data as needed
```

```
}
```

StopMatchBackfill()

Cancela una solicitud de reposición de emparejamiento activa que se creó con StartMatchBackfill(). Consulte también la acción del SDK de AWS StopMatchmaking(). Puede obtener más información sobre la característica de reposición de FlexMatch.

Sintaxis

```
GenericOutcome StopMatchBackfill (StopMatchBackfillRequest stopBackfillRequest);
```

Parámetros

StopMatchBackfillRequest

Un objeto StopMatchBackfillRequest que identifica el ticket de emparejamiento que se va a cancelar:

- ID del ticket asignado a la solicitud de reposición que se va a cancelar
- el creador de emparejamientos al que se envió la solicitud de reposición
- sesión de juego asociada a la solicitud de reposición

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
// 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()

Este método está obsoleto con la versión 4.0.1. En su lugar, el proceso del servidor debería llamar a ProcessEnding() una vez finalizada la sesión de juego.

Informa al servicio de Amazon GameLift de que el proceso del servidor ha finalizado la sesión de juego actual. Se llama a esta acción cuando el proceso del servidor permanece activo y listo para alojar una nueva sesión de juego. Solo debe llamarse una vez finalizado el procedimiento de finalización de la sesión de juego, ya que indica a Amazon GameLift que el proceso del servidor está disponible inmediatamente para alojar una nueva sesión de juego.

No se llamará a esta acción si el proceso del servidor se interrumpe una vez finalizada la sesión de juego. En su lugar, se llamará a ProcessEnding() para indicar que tanto la sesión de juego como el proceso del servidor están finalizando.

Sintaxis

```
GenericOutcome TerminateGameSession()
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo ilustra un proceso del servidor al final de una sesión de juego.

```
// 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()

Actualiza la capacidad de la sesión de juego actual para aceptar sesiones de jugador nuevas. Una sesión de juego se puede configurar para que acepte o deniegue todas las sesiones nuevas de los jugadores. (Consulte también la acción UpdateGameSession() en la Referencia de la API del servicio de Amazon GameLift).

Sintaxis

```
GenericOutcome UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy
 playerSessionPolicy)
```

Parámetros

newPlayerSessionPolicy

Valor de cadena que indica si la sesión de juego acepta jugadores nuevos.

Tipo: enum PlayerSessionCreationPolicy. Los valores válidos son:

- ACCEPT_ALL: se aceptan todas las sesiones de jugador nuevas.
- DENY_ALL: se rechazan todas las sesiones de jugador nuevas.

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo establece la política de participación en la sesión de juego actual para aceptar todos los jugadores.

```
var updatePlayerSessionCreationPolicyOutcomex =

 GameLiftServerAPI.UpdatePlayerSessionCreationPolicy(PlayerSessionCreationPolicy.ACCEPT_ALL);
```

Referencia del SDK del servidor de Amazon GameLift (C#): Tipos de datos

Puede utilizar esta referencia del SDK del servidor C# de Amazon GameLift como ayuda para preparar su juego multijugador para su uso con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

- Acciones
- Tipos de datos

LogParameters

Este tipo de datos se utiliza para identificar los archivos generados durante una sesión de juego que desea que Amazon GameLift cargue y almacene cuando finalice la sesión de juego. Esta información se comunicará al servicio de Amazon GameLift en una llamada a ProcessReady().

Contenido

logPaths

Lista de las rutas de directorio a archivos de registro del servidor de juegos que desea que Amazon GameLift almacene para futuros accesos. Estos archivos los genera un proceso del servidor durante cada sesión de juego; las rutas de archivo y los nombres se definen en el servidor de juegos y se almacenan en el directorio raíz de compilación de juegos. Las rutas del registro deben ser absolutas. Por ejemplo, si la compilación del juego almacena los logs de sesión de juego en una ruta del tipo `MyGame\sessionlogs\`, entonces la ruta de los logs sería `c:\game\MyGame\sessionLogs` (en una instancia Windows) o `/local/game/MyGame/sessionLogs` (en una instancia Linux).

Tipo: Lista<String>

Obligatorio: no

DescribePlayerSessionsRequest

Este tipo de datos se utiliza para especificar qué sesión o sesiones de jugador recuperar. Se puede utilizar de varias maneras: (1) proporcione un PlayerSessionId para solicitar una sesión de jugador específica; (2) proporcione un GameSessionId para solicitar todas las sesiones de jugador de la sesión de juego especificada; o (3) proporcione un PlayerId para solicitar todas las sesiones de jugador del jugador especificado. Para grandes recopilaciones de sesiones de jugador, utilice los parámetros de paginación para recuperar resultados como páginas secuenciales.

Contenido

GameSessionId

Identificador único de la sesión de juego. Use este parámetro para solicitar todas las sesiones de jugador de la sesión de juego especificada. El formato de ID de la sesión de juego es el siguiente: `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`. El valor de la **<**ID string**>** es una cadena de ID personalizada (si se especificó una cuando se creó la sesión de juego) o una cadena generada.

Tipo: String

Requerido: No

Límite

Número máximo de resultados a devolver. Use este parámetro con NextToken para obtener resultados en un conjunto de páginas secuenciales. Si se especifica un ID de sesión de jugador, este parámetro se ignora.

Tipo: entero

Obligatorio: no

NextToken

Token que indica el inicio de la siguiente página de resultados secuencial. Utilice el token devuelto con una llamada anterior a esta acción. Para especificar el inicio del conjunto de resultados, no indique ningún valor. Si se especifica un ID de sesión de jugador, este parámetro se ignora.

Tipo: String

Requerido: No

PlayerId

Identificador único de un jugador. Los ID de jugador los define el desarrollador. Consulte Generación de ID de jugador.

Tipo: String

Requerido: No

PlayerSessionId

Identificador único de una sesión de jugador.

Tipo: String

Requerido: No

PlayerSessionStatusFilter

Estado de la sesión de juego para filtrar los resultados. Los posibles estados de sesión de jugador son:

- RESERVED: se ha recibido la solicitud de sesión de jugador, pero el jugador aún no se ha conectado al proceso del servidor o aún no se ha validado.
- ACTIVE: el proceso del servidor ha validado el jugador y actualmente está conectado.
- COMPLETED: ha caído la conexión del jugador.
- TIMEDOUT: se ha recibido una solicitud de sesión de jugador, pero el jugador no se ha conectado y/o no se ha validado en el plazo de tiempo de espera (60 segundos).

Tipo: String

Requerido: No

ProcessParameters

Este tipo de datos contiene el conjunto de parámetros enviado a un servicio de Amazon GameLift en una llamada a ProcessReady().

Contenido

puerto

Es el número de puerto al que escucha el proceso del servidor para conexiones de jugador nuevas. El valor debe estar en el rango de puertos configurado para cualquier flota que implemente esa compilación de servidor de juegos. Este número de puerto se incluye en los objetos de sesión de juego y de jugador, que las sesiones de juego utilizan a la hora de conectarse a un proceso del servidor.

Tipo: entero

Obligatorio: sí

logParameters

Objeto con una lista de rutas de directorio a archivos de log de la sesión de juego.

Escriba: `Aws::GameLift::Server::LogParameters`

Obligatorio: sí

onStartGameSession

Nombre de la función de devolución de llamada que el servicio de Amazon GameLift invoca para activar una nueva sesión de juego. Amazon GameLift llama a esta función como respuesta a la solicitud de cliente CreateGameSession. La función de devolución de llamada toma un objeto GameSession (definido en la Referencia de la API del servicio de Amazon GameLift).

Escriba: `void OnStartGameSessionDelegate(GameSession gameSession)`

Obligatorio: sí

onProcessTerminate

Nombre de la función de devolución de llamada que el servicio de Amazon GameLift invoca para forzar el cierre del proceso de servidor. Después de llamar a esta función, Amazon GameLift espera cinco minutos hasta que el proceso de servidor se cierre y responde con una llamada a ProcessEnding() antes de cerrar el proceso de servidor.

Escriba: `void OnProcessTerminateDelegate()`

Obligatorio: sí

onHealthCheck

Nombre de la función de devolución de llamada que invoca el servicio de Amazon GameLift para solicitar un informe de estado del proceso de servidor. Amazon GameLift llama a esta función cada 60 segundos. Después de llamar a esta función, Amazon GameLift espera una respuesta durante 60 segundos y si no recibe ninguna, registra el proceso del servidor como en mal estado.

Escriba: `bool OnHealthCheckDelegate()`

Obligatorio: sí

onUpdateGameSession

Nombre de la función de devolución de llamada que el servicio de Amazon GameLift invoca para pasar un objeto de sesión de juego actualizado al proceso de servidor. Amazon GameLift llama a esta función cuando se ha procesado una solicitud de reposición de emparejamiento

para proporcionar datos actualizados de los emparejadores. Pasa un objeto GameSession, una actualización de estado (`updateReason`) y el ID del ticket de reposición de emparejamiento.

Escriba: `void OnUpdateGameSessionDelegate ( UpdateGameSession updateGameSession )`

Obligatorio: no

StartMatchBackfillRequest

Este tipo de datos se utiliza para enviar una solicitud de reposición de emparejamiento. La información se comunicará al servicio de Amazon GameLift en una llamada a StartMatchBackfill().

Contenido

GameSessionArn

Identificador único de la sesión de juego. El método del SDK GetGameSessionId() devuelve el identificador en formato de ARN.

Tipo: String

Obligatorio: sí

MatchmakingConfigurationArn

Identificador único, en forma de un ARN, que el creador de emparejamientos utiliza para esta solicitud. Para encontrar el creador de emparejamientos que se usó para crear la sesión de juego original, busque en el objeto de sesión de juego, en la propiedad de datos del creador de emparejamientos. Puede obtener más información sobre los datos del emparejador en Trabajo con datos del emparejador.

Tipo: String

Obligatorio: sí

Players

Un conjunto de datos que representa a todos los jugadores que están actualmente en la sesión de juego. El creador de emparejamientos utiliza esta información para buscar nuevos jugadores que son idóneos para los jugadores actuales. Para obtener una descripción del formato del objeto Player, consulte Guía de referencia de la API de Amazon GameLift. Para encontrar los atributos, ID y asignaciones de equipo del jugador, busque en el objeto de sesión de juego, en la propiedad

de datos del creador de emparejamientos. Si el creador de emparejamientos utiliza latencia, recopile la latencia actualizada para la región actual e inclúyala en los datos de cada jugador.

Tipo: Player[ ]

Obligatorio: sí

TicketId

Identificador único para un ticket de solicitud de emparejamiento o reposición de emparejamiento. Si no se proporciona ningún valor aquí, Amazon GameLift generará uno en forma de UUID. Use este identificador para realizar un seguimiento del estado del ticket de reposición de emparejamiento o cancelar la solicitud si es necesario.

Tipo: String

Requerido: No

StopMatchBackfillRequest

Este tipo de datos se utiliza para cancelar una solicitud de reposición de emparejamiento. La información se comunicará al servicio de Amazon GameLift en una llamada a StopMatchBackfill().

Contenido

GameSessionArn

Identificador único de sesión de juego asociado a la solicitud que se va a cancelar.

Tipo: String

Obligatorio: sí

MatchmakingConfigurationArn

Identificador único del creador de emparejamientos al que se envió esta solicitud.

Tipo: String

Obligatorio: sí

TicketId

Identificador único del ticket de solicitud de reposición que se va a cancelar.

Tipo: String

Obligatorio: sí

Referencia del SDK del servidor de Amazon GameLift para Go

Puede utilizar esta referencia del SDK del servidor de Go de Amazon GameLift como ayuda para preparar su juego multijugador para su uso con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

Temas

- Referencia GameLift del SDK (Go) del servidor Amazon: acciones
- Referencia GameLift del SDK (Go) del servidor Amazon: tipos de datos

Referencia GameLift del SDK (Go) del servidor Amazon: acciones

Puedes usar esta referencia del SDK del servidor Amazon GameLift Go como ayuda para preparar tu juego multijugador para usarlo con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

`GameLiftServerAPI.go` define las acciones del SDK del servidor Go.

Acciones

- GetSdkVersion()
- InitSDK()
- ProcessReady()
- ProcessEnding()
- ActivateGameSession()
- UpdatePlayerSessionCreationPolicy()
- GetGameSessionId()
- GetTerminationTime()
- AcceptPlayerSession()
- RemovePlayerSession()
- DescribePlayerSessions()
- StartMatchBackfill()
- StopMatchBackfill()

- GetComputeCertificate()
- GetFleetRoleCredentials()
- Destroy()

GetSdkVersion()

Devuelve el número de versión actual del SDK integrado en el proceso del servidor.

Sintaxis

```
func GetSdkVersion() (string, error)
```

Valor devuelto

Si funciona correctamente, devuelve la versión del SDK actual como una cadena. La cadena devuelta solo incluye el número de versión (por ejemplo, `5.0.0`). Si no funciona, devuelve un mensaje de error, como `common.SdkVersionDetectionFailed`.

Ejemplo

```
version, err := server.GetSdkVersion()
```

InitSDK()

Inicializa el Amazon GameLift SDK. Llama a este método en el lanzamiento antes de que se GameLift produzca cualquier otra inicialización relacionada con Amazon. Este método establece la comunicación entre el servidor y el GameLift servicio de Amazon.

Sintaxis

```
func InitSDK(params ServerParameters) error
```

Parámetros

ServerParameters

Para inicializar un servidor de juegos en una GameLift Anywhere flota de Amazon, construye un `ServerParameters` objeto con la siguiente información:

- La URL WebSocket utilizada para conectarte a tu servidor de juegos.
- El ID del proceso utilizado para alojar su servidor de juegos.

- El ID del proceso utilizado para alojar los procesos del servidor de juegos.
- El ID de la GameLift flota de Amazon que contiene tu ordenador de GameLift Anywhere Amazon.
- El token de autorización generado por la GameLift operación de Amazon.

Para inicializar un servidor de juegos en una flota de EC2 GameLift gestionada por Amazon, construye un `ServerParameters` objeto sin parámetros. Con esta llamada, el GameLift agente de Amazon configura el entorno informático y se conecta automáticamente al GameLift servicio de Amazon por usted.

Valor devuelto

Si funciona correctamente, devuelve el error `nil` para indicar que el proceso del servidor está listo para llamar a ProcessReady().

> **Note**
>
> Si las llamadas a `InitSDK()` no funcionan en las compilaciones de juegos implementadas en las flotas de Anywhere, compruebe el parámetro `ServerSdkVersion` que se utiliza al crear el recurso de compilación. Debe establecer este valor de forma explícita en la versión del SDK del servidor en uso. El valor predeterminado de este parámetro es 4.x, que no es compatible. Para resolver este problema, cree una compilación nueva e impleméntela en una flota nueva.

Ejemplo

GameLift AnywhereEjemplo de 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)
```

Ejemplo de EC2 GameLift gestionado por 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()

Notifica a Amazon de GameLift que el proceso del servidor está listo para albergar sesiones de juego. Llame a este método después de invocar InitSDK(). Se debe llamar a este método solo una vez por proceso.

Sintaxis

```
func ProcessReady(param ProcessParameters) error
```

Parámetros

ProcessParameters

Es un objeto ProcessParameters que comunica la siguiente información sobre el proceso del servidor:

- Los nombres de los métodos de devolución de llamada implementados en el código del servidor del juego que el GameLift servicio de Amazon invoca para comunicarse con el proceso del servidor.
- El número de puerto de escucha del servidor de proceso.
- El tipo de LogParameters datos que contiene la ruta a cualquier archivo específico de la sesión de juego que quieras que Amazon capture y GameLift almacene.

Valor devuelto

Devuelve un error con un mensaje de error si el método falla. Devuelve `nil` si el método se realiza correctamente.

Ejemplo

Este ejemplo ilustra las implementaciones tanto de la función de llamada ProcessReady() como de la función de delegación.

```
// Define the process parameters
processParams := ProcessParameters {
  OnStartGameSession: gameProcess.OnStartGameSession,
  OnUpdateGameSession: gameProcess.OnGameSessionUpdate,
  OnProcessTerminate: gameProcess.OnProcessTerminate,
  OnHealthCheck: gameProcess.OnHealthCheck,
  Port: port,
  LogParameters: LogParameters {    // logging and error example
    []string {"C:\\game\\logs", "C:\\game\\error"}
  }
}

err := server.ProcessReady(processParams)
```

ProcessEnding()

Notifica a Amazon de GameLift que el proceso del servidor está finalizando. Utiliza este método después de realizar todas las demás tareas de limpieza (incluido el cierre de la sesión de juego activa) y antes de finalizar el proceso. Según el resultado de `ProcessEnding()`, el proceso finaliza con éxito (0) o error (-1) y genera un evento de flota. Si el proceso termina con un error, se generará el evento de flota. `SERVER_PROCESS_TERMINATED_UNHEALTHY`

Sintaxis

```
func ProcessEnding() error
```

Valor devuelto

Devuelve un código de error 0 o un código de error definido.

Ejemplo

```
// operations to end game sessions and the server process
defer func() {
  err := server.ProcessEnding()
  server.Destroy()
  if err != nil {
```

```
    fmt.Println("ProcessEnding() failed. Error: ", err)
    os.Exit(-1)
  } else {
    os.Exit(0)
  }
}
```

ActivateGameSession()

Notifica a Amazon GameLift que el proceso del servidor ha activado una sesión de juego y ya está listo para recibir las conexiones de los jugadores. Se llama a esta acción como parte de la función de devolución de llamada `onStartGameSession()`, después de la inicialización de todas las sesiones de juego.

Sintaxis

```
func ActivateGameSession() error
```

Valor devuelto

Devuelve un error con un mensaje de error si el método falla.

Ejemplo

Este ejemplo muestra cómo se llama a `ActivateGameSession()` como parte de la función de delegación `onStartGameSession()`.

```
func OnStartGameSession(GameSession gameSession) {
  // game-specific tasks when starting a new game session, such as loading map
  // Activate when ready to receive players
  err := server.ActivateGameSession();
}
```

UpdatePlayerSessionCreationPolicy()

Actualiza la capacidad de la sesión de juego actual para aceptar sesiones de jugador nuevas. Una sesión de juego se puede configurar para que acepte o deniegue todas las sesiones nuevas de los jugadores.

Sintaxis

```
func UpdatePlayerSessionCreationPolicy(policy model.PlayerSessionCreationPolicy) error
```

Parámetros

playerSessionCreationPolítica

Valor de cadena que indica si la sesión de juego acepta jugadores nuevos.

Los valores válidos son:

- **`model.AcceptAll`**: se aceptan todas las sesiones de jugador nuevas.
- **`model.DenyAll`**: se rechazan todas las sesiones de jugador nuevas.

Valor devuelto

Devuelve un error con un mensaje de error si se produce un error.

Ejemplo

Este ejemplo establece la política de participación en la sesión de juego actual para aceptar todos los jugadores.

```
err := server.UpdatePlayerSessionCreationPolicy(model.AcceptAll)
```

GetGameSessionId()

Recupera el ID de la sesión de juego alojada por el proceso del servidor.

Sintaxis

```
func GetGameSessionID() (string, error)
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Si funciona correctamente, devuelve el ID de sesión del juego y el error nil. En el caso de los procesos inactivos que no se han activado aún con una sesión de juego, la llamada devuelve una cadena vacía y el error `nil`.

Ejemplo

```
gameSessionID, err := server.GetGameSessionID()
```

GetTerminationTime()

Devuelve la hora a la que está programada el cierre de un proceso de servidor, si hay una hora de terminación disponible. Un proceso de servidor realiza esta acción tras recibir una `onProcessTerminate()` llamada de Amazon GameLift. Amazon GameLift solicita `onProcessTerminate()` por los siguientes motivos:

- Cuando el proceso del servidor ha informado de un mal estado de salud o no ha respondido a Amazon GameLift.
- Al finalizar la instancia durante un evento de reducción vertical.
- Cuando se finaliza una instancia debido a la interrupción de una instancia de spot.

Sintaxis

```
func GetTerminationTime() (int64, error)
```

Valor devuelto

Si se ejecuta correctamente, devuelve la marca temporal en segundos en la que está previsto que el proceso del servidor se cierre y finalice el error `nil`. El valor es la hora de terminación expresado en ciclos transcurridos desde `0001 00:00:00`. Por ejemplo, el valor de la fecha y hora `2020-09-13 12:26:40 -000Z` es igual a `637355968000000000` ciclos. Si no hay una hora de terminación disponible, devuelve un mensaje de error.

Ejemplo

```
terminationTime, err := server.GetTerminationTime()
```

AcceptPlayerSession()

Notifica a Amazon GameLift que un jugador con el identificador de sesión de jugador especificado se ha conectado al proceso del servidor y necesita ser validado. Amazon GameLift verifica que el identificador de sesión del jugador sea válido. Una vez validada la sesión del jugador, Amazon GameLift cambia el estado de la ranura del jugador de `RESERVED` a`ACTIVE`.

Sintaxis

```
func AcceptPlayerSession(playerSessionID string) error
```

Parámetros

`playerSessionId`

ID único emitido por Amazon GameLift cuando se crea una nueva sesión de jugador.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

En este ejemplo, se administra una solicitud de conexión que incluye la validación y el rechazo de los ID de sesión de los jugadores no válidos.

```
func ReceiveConnectingPlayerSessionID(conn Connection, playerSessionID string) {
    err := server.AcceptPlayerSession(playerSessionID)
    if err != nil {
        connection.Accept()
    } else {
        connection.Reject(err.Error())
    }
}
```

RemovePlayerSession()

Notifica a Amazon GameLift que un jugador se ha desconectado del proceso del servidor. En respuesta, Amazon GameLift cambia el espacio del jugador para que esté disponible.

Sintaxis

```
func RemovePlayerSession(playerSessionID string) error
```

Parámetros

`playerSessionId`

ID único emitido por Amazon GameLift cuando se crea una nueva sesión de jugador.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
err := server.RemovePlayerSession(playerSessionID)
```

DescribePlayerSessions()

Recupera datos de sesión de jugador, incluida la configuración, los metadatos de la sesión y los datos de jugador. Utilice este método para obtener información sobre los siguientes elementos:

- Una sesión para un jugador
- Todas las sesiones del jugador en una sesión de juego
- Todas las sesiones de jugador están asociadas a un único ID de jugador

Sintaxis

```
func DescribePlayerSessions(req request.DescribePlayerSessionsRequest)
 (result.DescribePlayerSessionsResult, error) {
 return srv.describePlayerSessions(&req)
}
```

Parámetros

DescribePlayerSessionsRequest

Es un objeto `DescribePlayerSessionsRequest` que describe las sesiones de jugador a recuperar.

Valor devuelto

Si funciona correctamente, devuelve un objeto `DescribePlayerSessionsResult` que contiene un conjunto de objetos de sesión de jugador que se ajusta a los parámetros de la solicitud.

Ejemplo

En este ejemplo se solicitan todas las sesiones de jugador conectadas activamente a una sesión de juego específica. Al omitir NextTokeny establecer el valor límite en 10, Amazon GameLift devuelve los registros de las sesiones de los primeros 10 jugadores que coincidan con la solicitud.

```
// 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()

Envía una solicitud para encontrar nuevos jugadores para ranuras abiertas en una sesión de juego creada con FlexMatch. Para obtener más información, consulta la función FlexMatch de relleno.

Esta acción es asíncrona. Si se emparejan nuevos jugadores, Amazon GameLift proporciona datos actualizados de los emparejadores mediante la función de devolución de llamada. `OnUpdateGameSession()`

Un proceso del servidor solo puede tener una solicitud de reposición de emparejamiento activa a la vez. Para enviar una nueva solicitud, en primer lugar llame a StopMatchBackfill() para cancelar la solicitud original.

Sintaxis

```
func StartMatchBackfill(req request.StartMatchBackfillRequest)
 (result.StartMatchBackfillResult, error)
```

Parámetros

StartMatchBackfillRequest

Un StartMatchBackfillRequest objeto comunica la siguiente información:

- Un ID de ticket que se asignará a la solicitud de reposición. Esta información es opcional; si no se proporciona ningún identificador, Amazon GameLift generará uno.
- El creador de emparejamientos al que se enviará la solicitud. El ARN de configuración completo es obligatorio. Este valor se encuentra en los datos del emparejador de la sesión de juego.
- El ID de la sesión de juego que se va a reponer.
- Datos del emparejador disponibles para los jugadores actuales de la sesión de juego.

Valor devuelto

Devuelve un objeto `StartMatchBackfillResult` con el ID del ticket de reposición de emparejamiento o un error con un mensaje de error.

Ejemplo

```
// form the request
startBackfillRequest := request.NewStartMatchBackfill()
startBackfillRequest.RequestID = "1111aaaa-22bb-33cc-44dd-5555eeee66ff"          //
 optional
startBackfillRequest.MatchmakingConfigurationArn = "arn:aws:gamelift:us-
west-2:111122223333:matchmakingconfiguration/MyMatchmakerConfig"
var matchMaker model.MatchmakerData
if err := matchMaker.UnmarshalJSON([]byte(gameSession.MatchmakerData)); err != nil {

    return
}
startBackfillRequest.Players = matchMaker.Players
res, err := server.StartMatchBackfill(startBackfillRequest)

// Implement callback function for backfill
func OnUpdateGameSession(myGameSession model.GameSession) {
    // game-specific tasks to prepare for the newly matched players and update
 matchmaker data as needed
}
```

StopMatchBackfill()

Cancela una solicitud de reposición de emparejamiento activa. Para obtener más información, consulta la función de FlexMatch relleno.

Sintaxis

```
func StopMatchBackfill(req request.StopMatchBackfillRequest) error
```

Parámetros

StopMatchBackfillRequest

Un StopMatchBackfillRequest objeto que identifica el billete de emparejamiento que se va a cancelar:

- ID del ticket que se asignará a la solicitud de reposición
- El emparejador al que se envió la solicitud de reposición
- La sesión de juego asociada a la solicitud de reposición.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
stopBackfillRequest := request.NewStopMatchBackfill()  // Use this function to create
 request
stopBackfillRequest.TicketID = "1111aaaa-22bb-33cc-44dd-5555eeee66ff"
stopBackfillRequest.MatchmakingConfigurationArn = "arn:aws:gamelift:us-
west-2:111122223333:matchmakingconfiguration/MyMatchmakerConfig"

//error
err := server.StopMatchBackfill(stopBackfillRequest)
```

GetComputeCertificate()

Recupera la ruta al certificado TLS utilizado para cifrar la conexión de red entre el servidor de juegos y el cliente de juego. Puedes usar la ruta del certificado al registrar tu dispositivo informático en una GameLift Anywhere flota de Amazon. Para obtener más información, consulte RegisterCompute.

Sintaxis

```
func GetComputeCertificate() (result.GetComputeCertificateResult, error)
```

Valor devuelto

Devuelve un objeto `GetComputeCertificateResult` que contiene los siguientes elementos:

- CertificatePath: La ruta al certificado TLS de su recurso informático. Cuando se utiliza una flota GameLift gestionada por Amazon, esta ruta contiene:
  - `certificate.pem`: el certificado del usuario final. La cadena de certificados completa es la combinación del `certificateChain.pem` adjunto a este certificado.
  - `certificateChain.pem`: la cadena de certificados que contiene el certificado raíz y los certificados intermedios.

- `rootCertificate.pem`: el certificado raíz.
- `privateKey.pem`: la clave privada del certificado del usuario final.
- ComputeName: el nombre de su recurso informático.

Ejemplo

```
tlsCertificate, err := server.GetFleetRoleCredentials(getFleetRoleCredentialsRequest)
```

GetFleetRoleCredentials()

Recupera las credenciales del rol de servicio que creaste para extender los permisos a tu otro Servicios de AWS rol en Amazon GameLift. Estas credenciales permiten que su servidor de juegos utilice sus recursos de AWS. Para obtener más información, consulte Configurar un rol de servicio de IAM para Amazon GameLift.

Sintaxis

```
func GetFleetRoleCredentials(
  req request.GetFleetRoleCredentialsRequest,
) (result.GetFleetRoleCredentialsResult, error) {
  return srv.getFleetRoleCredentials(&req)
}
```

Parámetros

GetFleetRoleCredentialsRequest

Credenciales de rol que amplían el acceso limitado a sus recursos de AWS al servidor de juegos.

Valor devuelto

Devuelve un objeto `GetFleetRoleCredentialsResult` que contiene los siguientes elementos:

- AssumedRoleUserArn - El nombre de recurso de Amazon (ARN) del usuario al que pertenece la función de servicio.
- AssumedRoleId - El ID del usuario al que pertenece el rol de servicio.
- AccessKeyId - El ID de la clave de acceso para autenticar y proporcionar acceso a sus AWS recursos.

- SecretAccessKey - El identificador de la clave de acceso secreta para la autenticación.
- SessionToken - Un token para identificar la sesión activa actual que interactúa con tus AWS recursos.
- Vencimiento: el tiempo que queda hasta que caduquen las credenciales de la sesión.

Ejemplo

```
// form the customer credentials request
getFleetRoleCredentialsRequest := request.NewGetFleetRoleCredentials()
getFleetRoleCredentialsRequest.RoleArn = "arn:aws:iam::123456789012:role/service-role/
exampleGameLiftAction"

credentials, err := server.GetFleetRoleCredentials(getFleetRoleCredentialsRequest)
```

Destroy()

Libera de la memoria el SDK del servidor de GameLift juegos de Amazon. Como práctica recomendada, llame a este método después de `ProcessEnding()` y antes de finalizar el proceso. Si utilizas una flota de Anywhere y no vas a finalizar los procesos del servidor después de cada sesión de juego, llama `Destroy()` y, `InitSDK()` a continuación, reinicializa antes de notificar a Amazon GameLift que el proceso está listo para organizar una sesión de juego con ella. `ProcessReady()`

Sintaxis

```
func Destroy() error {
 return srv.destroy()
}
```

Valor devuelto

Devuelve un error con un mensaje de error si el método falla.

Ejemplo

```
// 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)
  }
}
```

Referencia GameLift del SDK (Go) del servidor Amazon: tipos de datos

Puedes usar esta referencia del SDK del servidor Amazon GameLift Go como ayuda para preparar tu juego multijugador para usarlo con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

Tipos de datos

- LogParameters
- ProcessParameters
- UpdateGameSession
- GameSession
- ServerParameters
- StartMatchBackfillRequest
- Jugador
- DescribePlayerSessionsRequest
- StopMatchBackfillRequest
- GetFleetRoleCredentialsRequest

LogParameters

Un objeto que identifica los archivos generados durante una sesión de juego que quieres que Amazon GameLift suba y almacene una vez finalizada la sesión de juego. El servidor del juego proporciona `LogParameters` a Amazon GameLift como parte de un `ProcessParameters` objeto en una ProcessReady() llamada.

Propiedades	Descripción

`LogPaths`	La lista de rutas de directorio a los archivos de registro del servidor de juegos que quieres que Amazon almacene GameLift para acceder a ellos en el futuro. El proceso del servidor genera esos archivos durante una sesión de juego. Defina los nombres y las rutas de los archivos en el servidor de juegos y almacénelos en el directorio raíz de compilación del juego. Las rutas del registro deben ser absolutas. Por ejemplo, si la compilación del juego almacena los registros de sesión de juego en una ruta del tipo `MyGame\sessionLogs\` , la ruta sería `c:\game\MyGame\sessionLogs` en una instancia de Windows. Tipo: `[]string` Obligatorio: no

ProcessParameters

Objeto que describe la comunicación entre un proceso de servidor y Amazon GameLift. El proceso del servidor proporciona esta información a Amazon GameLift con una llamada aProcessReady().

Propiedades	Descripción
`LogParameters`	Un objeto con rutas de directorio a los archivos que se generan durante una sesión de juego. Amazon GameLift copia y almacena los archivos para acceder a ellos en el futuro. Tipo: `LogParameters` Obligatorio: no
`OnHealthCheck`	La función de devolución de llamada que Amazon GameLift invoca para solicitar un informe de estado de salud al proceso del servidor. Amazon GameLift llama a esta función cada 60 segundos y espera 60 segundos para recibir una respuesta. El proceso del servidor devuelve `TRUE` si está en buen estado y `FALSE` si no. Si no se devuelve ninguna respuesta, Amazon GameLift registra el proceso del servidor como incorrecto. Tipo: `OnHealthCheck func() bool`

	Obligatorio: no
`OnProcessTerminate`	La función de devolución de llamada que Amazon GameLift invoca para forzar el cierre del proceso del servidor. Tras llamar a esta función, Amazon GameLift espera 5 minutos a que se cierre el proceso del servidor y responde con una ProcessEnding() llamada antes de cerrar el proceso del servidor. Tipo: `OnProcessTerminate func()` Obligatorio: sí
`OnStartGameSession`	La función de devolución de llamada que Amazon GameLift invoca para pasar un objeto de sesión de juego actualizado al proceso del servidor. Amazon GameLift llama a esta función cuando se ha procesado una solicitud de relleno de partidas para proporcionar datos actualizados de los emparejadores. Transmite un GameSessionobjeto, una actualización de estado (`updateReason`) y el identificador del ticket de relleno del partido. Tipo: `OnStartGameSession func (model.GameSession )` Obligatorio: sí
`OnUpdateGameSession`	La función de devolución de llamada que Amazon GameLift invoca para pasar la información actualizada de la sesión de juego al servidor es el proceso. Amazon GameLift llama a esta función después de procesar una solicitud de relleno de partidas para proporcionar datos actualizados de los emparejadores. Tipo: `OnUpdateGameSession func (model.UpdateGameSession)` Obligatorio: no

`Port`	El número de puerto en el que escucha el proceso del servidor para recibir conexiones de jugador nuevas. El valor debe estar en el rango de puertos configurado para cualquier flota que implemente esa compilación de servidor de juegos. Este número de puerto se incluye en los objetos de sesión de juego y de jugador, que las sesiones de juego utilizan a la hora de conectarse a un proceso del servidor. Tipo: `int` Obligatorio: sí

UpdateGameSession

Las actualizaciones en un objeto de sesión de juego, que incluye el motivo por el que se actualizó la sesión de juego y el ID del ticket de reposición correspondiente si la reposición se utiliza para reponer las sesiones de los jugadores en la sesión de juego.

Propiedades	Descripción
GameSession	Un GameSessionobjeto definido por la GameLift API de Amazon. El objeto `GameSession` contiene propiedades que describen una sesión de juego. Tipo: `GameSession GameSession()` Obligatorio: sí
UpdateReason	El motivo por el que se actualiza la sesión de juego. Tipo: `UpdateReason UpdateReason()` Obligatorio: sí
BackfillTicketId	El ID de ticket de reposición que intenta actualizar la sesión de juego. Tipo: `String` Obligatorio: no

GameSession

Los detalles de una sesión de juego.

Propiedades	Descripción
GameSessionId	Un identificador único de la sesión de juego. Un nombre de recurso de Amazon (ARN) de sesión de juego tiene el siguiente formato: `arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token>` . Tipo: `String` Obligatorio: no
Nombre	Una etiqueta descriptiva de la sesión de juego. Tipo: `String` Obligatorio: no
FleetId	Un identificador único para la flota en la que se ejecuta la sesión de juego. Tipo: `String` Obligatorio: no
MaximumPlayerSessionCount	El número máximo de conexiones de jugadores a la sesión de juego. Tipo: `Integer` Obligatorio: no
Puerto	El número de puerto de la sesión de juego. Para conectarse a un servidor de GameLift juegos de Amazon, una aplicación necesita tanto la dirección IP como el número de puerto. Tipo: `Integer` Obligatorio: no

Propiedades	Descripción
IpAddress	La dirección IP de la sesión de juego. Para conectarse a un servidor de GameLift juegos de Amazon, una aplicación necesita tanto la dirección IP como el número de puerto. Tipo: `String` Obligatorio: no
GameSessionData	Un conjunto de propiedades de sesión de juego personalizadas, formateadas como un valor de una sola cadena. Tipo: `String` Obligatorio: no
MatchmakerData	La información sobre el proceso de emparejamiento que se utilizó para crear la sesión de juego, en sintaxis JSON, con formato como cadena. Además de la configuración de emparejamiento utilizada, contiene datos sobre todos los jugadores asignados al emparejamiento, incluidos los atributos de los jugadores y las asignaciones de los equipos. Tipo: `String` Obligatorio: no
GameProperties	Un conjunto de propiedades personalizadas de una sesión de juego, con formato como pares clave-valor. Estas propiedades se pasan a una solicitud de iniciar una nueva sesión de juego. Tipo: `map[string] string` Obligatorio: no

Propiedades	Descripción
DnsName	El identificador de DNS asignado a la instancia que ejecuta la sesión de juego. Los valores tienen formato siguiente: • Flotas habilitadas para TLS: `<unique identifier>.<region identifier>.amazongamelift.com` . • Flotas no habilitadas para TLS: `ec2-<unique identifier>.compute.amazonaws.com` . Cuando se conecte a una sesión de juego que se ejecute en una flota habilitada de TLS, debe utilizar el nombre de DNS, no la dirección IP. Tipo: `String` Obligatorio: no

ServerParameters

Información utilizada para mantener la conexión entre un GameLift Anywhere servidor de Amazon y el GameLift servicio de Amazon. Esta información se utiliza al inicializar nuevos procesos de servidor con InitSDK(). Para los servidores alojados en instancias EC2 GameLift gestionadas por Amazon, utilice un objeto vacío.

Propiedades	Descripción
`WebSocketURL`	`GameLiftServerSdkEndpoint` Amazon GameLift regresa cuando `RegisterCompute` buscas un recurso GameLift Anywhere informático de Amazon. Tipo: `string` Obligatorio: sí
`ProcessID`	Un identificador único registrado en el proceso del servidor que aloja el juego. Tipo: `string`

Propiedades	Descripción
	Obligatorio: sí
`HostID`	El identificador único del recurso informático que aloja el nuevo proceso del servidor. El `HostID` es el `ComputeName` utilizado cuando registró el recurso informático. Para obtener más información, consulte RegisterCompute. Tipo: `string` Obligatorio: sí
`FleetID`	El identificador único de la flota en la que está registrado el recurso informático. Para obtener más información, consulte RegisterCompute. Tipo: `string` Obligatorio: sí
`AuthToken`	El token de autenticación generado por Amazon GameLift que autentica tu servidor en Amazon GameLift. Para obtener más información, consulte GetComputeAuthToken. Tipo: `string` Obligatorio: sí

StartMatchBackfillRequest

Información utilizada para crear una solicitud de reposición de emparejamiento. El servidor del juego comunica esta información a Amazon GameLift en una StartMatchBackfill() llamada.

Propiedades	Descripción
GameSessionArn	El identificador único de la sesión de juego. El `GetGameSessionId` de la operación de la API devuelve el identificador en formato de ARN. Tipo: `String`

Propiedades	Descripción
	Obligatorio: sí
MatchmakingConfigurationArn	El identificador único, en forma de ARN, que el emparejador utiliza para esta solicitud. El ARN del emparejador para la sesión de juego original se encuentra en el objeto de sesión de juego en la propiedad de datos del emparejador. Para obtener más información sobre los datos del emparejador, consulte Trabajo con datos del emparejador. Tipo: `String` Obligatorio: sí
Players	Un conjunto de datos que representa a todos los jugadores que están actualmente en la sesión de juego. El creador de emparejamientos utiliza esta información para buscar nuevos jugadores que son idóneos para los jugadores actuales. Tipo: `[]model.Player` Obligatorio: sí
TicketId	El identificador único para un ticket de solicitud de emparejamiento o reposición de emparejamiento. Si no proporcionas un valor, Amazon GameLift generará uno. Use este identificador para realizar un seguimiento del estado del ticket de reposición de emparejamiento o cancelar la solicitud si es necesario. Tipo: `String` Obligatorio: no

Jugador

El objeto que representa a un jugador en el emparejamiento. Cuando se inicia una solicitud de emparejamiento, un jugador tiene un ID de jugador, atributos y, posiblemente, datos de latencia. Amazon GameLift añade la información del equipo después de que se haya disputado un partido.

Propiedades	Descripción
LatencyInMS	Un conjunto de valores expresados en milisegundos que indican la cantidad de latencia que experimenta un jugador cuando se conecta a una ubicación. Si se utiliza esta propiedad, el jugador solo se empareja con las ubicaciones que aparecen. Si un creador de emparejamientos tiene una regla que evalúa la latencia de los jugadores, estos deben informar de la latencia para ser emparejados. Tipo: `map[string] int` Obligatorio: no
PlayerAttributes	Una colección de pares de clave-valor que contienen información del jugador para su uso en el emparejamiento. Las claves de atributos del jugador deben coincidir con las PlayerAttributes utilizadas en un conjunto de reglas de emparejamiento. Para obtener más información sobre los atributos de los jugadores, consulte AttributeValue. Tipo: `map[string] AttributeValue` Obligatorio: no
PlayerId	Un identificador único de un jugador. Tipo: `String` Obligatorio: no
Equipo	El nombre del equipo al que está asignado el jugador en un emparejamiento. Defina el nombre del equipo se define en el conjunto de reglas de emparejamiento. Tipo: `String` Obligatorio: no

DescribePlayerSessionsRequest

Un objeto que especifica las sesiones de jugador que recuperar. El proceso del servidor proporciona esta información con una DescribePlayerSessions() llamada a Amazon GameLift.

Propiedades	Descripción
`GameSessionID`	Un identificador único de la sesión de juegos. Use este parámetro para solicitar todas las sesiones de jugador de la sesión de juego especificada. El formato de ID de sesión de juego es `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>` . El `GameSessionID` es una cadena de ID personalizada o una cadena generada. Tipo: `String` Obligatorio: no
`PlayerSessionID`	El identificador único de una sesión de jugador. Utilice este parámetro para solicitar una sesión de jugador específica. Tipo: `String` Obligatorio: no
`PlayerID`	El identificador único de un jugador. Utilice este parámetro para solicitar todas las sesiones de jugador para un jugador específico. Consulte Generación de ID de jugador. Tipo: `String` Obligatorio: no
`PlayerSessionStatusFilter`	El estado de la sesión de jugador para filtrar los resultados. Los posibles estados de sesión de jugador son los siguientes: • RESERVADO: se ha recibido la solicitud de sesión de jugador, pero el jugador aún no se ha conectado al proceso del servidor o aún no se ha validado.

Propiedades	Descripción
	• ACTIVO: el proceso del servidor ha validado el jugador y está conectado. • COMPLETO: la conexión del jugador se ha caído. • TIEMPO DE ESPERA AGOTADO: se ha recibido una solicitud de sesión de jugador, pero el jugador no se ha conectado y/o no se ha validado en el plazo de tiempo de espera (60 segundos). Tipo: `String` Obligatorio: no
`NextToken`	El token que indica el inicio de la siguiente página de resultados. Para especificar el inicio del conjunto de resultados, no indique ningún valor. Si se especifica un ID de sesión de jugador, este parámetro se ignora. Tipo: `String` Obligatorio: no
`Limit`	El número máximo de resultados que devolver. Si se especifica un ID de sesión de jugador, este parámetro se ignora. Tipo: `int` Obligatorio: no

StopMatchBackfillRequest

Información utilizada para cancelar una solicitud de reposición de emparejamiento. El servidor del juego comunica esta información al GameLift servicio de Amazon en una StopMatchBackfill() llamada.

Propiedades	Descripción
GameSessionArn	El identificador único de sesión de juego de la solicitud que se va a cancelar. Tipo: `string`

Propiedades	Descripción
	Obligatorio: no
MatchmakingConfigurationArn	El identificador único del emparejador al que se envió esta solicitud. Tipo: `string` Obligatorio: no
TicketId	El identificador único del ticket de solicitud de reposición que se va a cancelar. Tipo: `string` Obligatorio: no

GetFleetRoleCredentialsRequest

Las credenciales de rol que amplían el acceso limitado a sus recursos de AWS al servidor de juegos. Para obtener más información, consulte Configurar un rol de servicio de IAM para Amazon GameLift.

Propiedades	Descripción
RoleArn	El ARN del rol de servicio que amplía el acceso limitado a sus recursos de AWS. Tipo: `string` Obligatorio: sí
RoleSessionName	El nombre de la sesión que describe el uso de las credenciales del rol. Tipo: `string` Obligatorio: sí

Referencia del SDK del servidor de Amazon GameLift para Unreal Engine

Esta referencia del servidor de Amazon GameLift puede ayudarle a preparar los proyectos de juego de Unreal Engine para utilizarlos con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

Esta API se define en `GameLiftServerSDK.h` y `GameLiftServerSDKModels.h`.

Para configurar el complemento Unreal Engine y ver ejemplos de código, consulte Integre Amazon GameLift en un proyecto de Unreal Engine.

Temas

- Referencia del SDK del servidor de Unreal Engine de Amazon GameLift 5.x
- Referencia del SDK del servidor de Unreal Engine de Amazon GameLift 3.x

Referencia del SDK del servidor de Unreal Engine de Amazon GameLift 5.x

Puede utilizar esta referencia del SDK del servidor de Unreal Engine de Amazon GameLift 5.x como ayuda para preparar su juego multijugador para su uso con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos y para obtener información sobre el uso del complemento del servidor del SDK de Unreal, consulte Integre Amazon GameLift en un proyecto de Unreal Engine.

Temas

- Referencia 5.x GameLift del SDK del servidor Amazon (Unreal): Acciones
- Referencia GameLift del SDK (Unreal) del servidor Amazon: tipos de datos

Referencia 5.x GameLift del SDK del servidor Amazon (Unreal): Acciones

Puedes usar esta referencia del SDK del servidor Amazon GameLift Unreal como ayuda para preparar tu juego multijugador para usarlo con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos y para obtener información sobre el uso del complemento del servidor del SDK de Unreal, consulte Integre Amazon GameLift en un proyecto de Unreal Engine.

Acciones

- GetSdkVersion()

- InitSDK()
- InitSDK()
- ProcessReady()
- ProcessEnding()
- ActivateGameSession()
- UpdatePlayerSessionCreationPolicy()
- GetGameSessionId()
- GetTerminationTime()
- AcceptPlayerSession()
- RemovePlayerSession()
- DescribePlayerSessions()
- StartMatchBackfill()
- StopMatchBackfill()
- GetComputeCertificate()
- GetFleetRoleCredentials()

> **Note**
>
> En este tema se describe la API de Amazon GameLift C++ que puede usar al compilar para Unreal Engine. En concreto, esta documentación se aplica al código que se compila con la opción `-DBUILD_FOR_UNREAL=1`.

GetSdkVersion()

Devuelve el número de versión actual del SDK integrado en el proceso del servidor.

Sintaxis

```
FGameLiftStringOutcome GetSdkVersion();
```

Valor devuelto

Si funciona correctamente, devuelve la versión del SDK actual como objeto the section called “F GameLiftStringOutcome”. El objeto devuelto incluye el número de versión (por ejemplo, `5.0.0`). Si no funciona, devuelve un mensaje de error.

Ejemplo

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =
 Aws::GameLift::Server::GetSdkVersion();
```

InitSDK()

Inicializa el Amazon GameLift SDK para una flota de EC2 gestionada. Llama a este método en el momento del lanzamiento, antes de que GameLift se produzca cualquier otra inicialización relacionada con Amazon. Este método lee los parámetros del servidor del entorno anfitrión para configurar la comunicación entre el servidor y el GameLift servicio de Amazon.

Sintaxis

```
FGameLiftGenericOutcome InitSDK()
```

Valor devuelto

Si funciona correctamente, devuelve un objeto `InitSdkOutcome` e indica que el proceso del servidor está listo para llamar a ProcessReady().

Ejemplo

```
//Call InitSDK to establish a local connection with the GameLift agent to enable
 further communication.
FGameLiftGenericOutcome initSdkOutcome = gameLiftSdkModule->InitSDK();
```

InitSDK()

Inicializa el Amazon GameLift SDK para una Anywhere flota. Llama a este método en el momento del lanzamiento, antes de que GameLift se produzca cualquier otra inicialización relacionada con Amazon. Este método requiere parámetros de servidor explícitos para configurar la comunicación entre el servidor y el GameLift servicio de Amazon.

Sintaxis

```
FGameLiftGenericOutcome InitSDK(serverParameters)
```

Parámetros

F ServerParameters

Para inicializar un servidor de juegos en una GameLift Anywhere flota de Amazon, construye un `ServerParameters` objeto con la siguiente información:

- La URL WebSocket utilizada para conectarte a tu servidor de juegos.
- El ID del proceso utilizado para alojar su servidor de juegos.
- El ID del proceso utilizado para alojar los procesos del servidor de juegos.
- El ID de la GameLift flota de Amazon que contiene tu ordenador de GameLift Anywhere Amazon.
- El token de autorización generado por la GameLift operación de Amazon.

Valor devuelto

Si funciona correctamente, devuelve un objeto `InitSdkOutcome` e indica que el proceso del servidor está listo para llamar a ProcessReady().

> **Note**
>
> Si las llamadas a `InitSDK()` no funcionan en las compilaciones de juegos implementadas en las flotas de Anywhere, compruebe el parámetro `ServerSdkVersion` que se utiliza al crear el recurso de compilación. Debe establecer este valor de forma explícita en la versión del SDK del servidor en uso. El valor predeterminado de este parámetro es 4.x, que no es compatible. Para resolver este problema, cree una compilación nueva e impleméntela en una flota nueva.

Ejemplo

```
//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()

Notifica a Amazon de GameLift que el proceso del servidor está listo para albergar sesiones de juego. Llame a este método después de invocar InitSDK(). Se debe llamar a este método solo una vez por proceso.

Sintaxis

```
GenericOutcome ProcessReady(const Aws::GameLift::Server::ProcessParameters
&processParameters);
```

Parámetros

processParameters

Un objeto F ProcessParameters que comunica la siguiente información sobre el proceso del servidor:

- Nombres de los métodos de devolución de llamada implementados en el código del servidor del juego que el GameLift servicio de Amazon invoca para comunicarse con el proceso del servidor.
- Número de puerto de escucha del servidor de proceso.
- Ruta a cualquier archivo específico de la sesión de juego que quieras que Amazon capture y GameLift almacene.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo ilustra las implementaciones tanto de la función de llamada ProcessReady() como de la función de delegación.

```
//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()

Notifica a Amazon de GameLift que el proceso del servidor está finalizando. Utiliza este método después de realizar todas las demás tareas de limpieza (incluido el cierre de la sesión de juego activa) y antes de finalizar el proceso. Según el resultado de `ProcessEnding()`, el proceso finaliza con éxito (0) o error (-1) y genera un evento de flota. Si el proceso termina con un error, el evento de flota generado es `SERVER_PROCESS_TERMINATED_UNHEALTHY`.

Sintaxis

```
FGameLiftGenericOutcome ProcessEnding()
```

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
//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()

Notifica a Amazon GameLift que el proceso del servidor ha activado una sesión de juego y ya está listo para recibir las conexiones de los jugadores. Esta acción debe llamarse como parte de la función de devolución de llamada `onStartGameSession()`, después de la inicialización de todas las sesiones de juego.

Sintaxis

```
FGameLiftGenericOutcome ActivateGameSession()
```

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo muestra cómo se llama a `ActivateGameSession()` como parte de la función de delegación `onStartGameSession()`.

```
//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()

Actualiza la capacidad de la sesión de juego actual para aceptar sesiones de jugador nuevas. Una sesión de juego se puede configurar para que acepte o deniegue todas las sesiones nuevas de los jugadores.

Sintaxis

```
FGameLiftGenericOutcome UpdatePlayerSessionCreationPolicy(EPlayerSessionCreationPolicy
 policy)
```

Parámetros

playerCreationSessionPolítica

Valor de cadena que indica si la sesión de juego acepta jugadores nuevos.

Los valores válidos son:

- ACCEPT_ALL: se aceptan todas las sesiones de jugador nuevas.
- DENY_ALL: se rechazan todas las sesiones de jugador nuevas.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Este ejemplo establece la política de participación en la sesión de juego actual para aceptar todos los jugadores.

```
FGameLiftGenericOutcome outcome = gameLiftSdkModule-
>UpdatePlayerSessionCreationPolicy(Aws::GameLift::Model::EPlayerSessionCreationPolicy::ACCEPT_A
```

GetGameSessionId()

Recupera el ID de la sesión de juego alojada por el proceso del servidor.

En el caso de los procesos inactivos que no se activan con una sesión de juego, la llamada devuelve the section called “F GameLiftError”.

Sintaxis

```
FGameLiftStringOutcome GetGameSessionId()
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Si funciona correctamente, devuelve el ID de sesión del juego como objeto the section called “F GameLiftStringOutcome”. Si no funciona, devuelve un mensaje de error.

En el caso de los procesos inactivos que no se activan con una sesión de juego, la llamada devuelve `Success=True` y `GameSessionId=""`.

Ejemplo

```
//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()

Devuelve la hora a la que está programada el cierre de un proceso de servidor, si hay una hora de terminación disponible. Un proceso de servidor toma medidas después de recibir una `onProcessTerminate()` llamada de Amazon GameLift. Amazon GameLift solicita `onProcessTerminate()` por los siguientes motivos:

- Cuando el proceso del servidor ha informado de un mal estado de salud o no ha respondido a Amazon GameLift.
- Al finalizar la instancia durante un evento de reducción vertical.
- Cuando se finaliza una instancia debido a la interrupción de una instancia de spot.

Sintaxis

```
AwsDateTimeOutcome GetTerminationTime()
```

Valor devuelto

Si el proceso se realiza correctamente, devuelve la hora de terminación como un objeto `AwsDateTimeOutcome`. El valor es la hora de terminación expresado en ciclos transcurridos desde `0001 00:00:00`. Por ejemplo, el valor de la fecha y hora `2020-09-13 12:26:40 -000Z` es igual a `637355968000000000` ciclos. Si no hay una hora de terminación disponible, devuelve un mensaje de error.

Si el proceso no ha recibido ninguna devolución de llamada `ProcessParameters.OnProcessTerminate()`, se devuelve un mensaje de error. Para obtener más información sobre cómo cerrar un proceso de servidor, consulte Respuesta a una notificación de cierre del proceso del servidor.

Ejemplo

```
AwsDateTimeOutcome TermTimeOutcome = gameLiftSdkModule->GetTerminationTime();
```

AcceptPlayerSession()

Notifica a Amazon GameLift que un jugador con el identificador de sesión de jugador especificado se ha conectado al proceso del servidor y necesita ser validado. Amazon GameLift verifica que el identificador de sesión del jugador sea válido. Una vez validada la sesión del jugador, Amazon GameLift cambia el estado del espacio del jugador de RESERVADO a ACTIVO.

Sintaxis

```
FGameLiftGenericOutcome AcceptPlayerSession(const FString& playerSessionId)
```

Parámetros

playerSessionId

ID único emitido por Amazon GameLift cuando se crea una nueva sesión de jugador.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

En este ejemplo, se administra una solicitud de conexión que incluye la validación y el rechazo de los ID de sesión de los jugadores no válidos.

```
bool GameLiftManager::AcceptPlayerSession(const FString& playerSessionId, const
 FString& playerId)
{
  #if WITH_GAMELIFT
  UE_LOG(GameServerLog, Log, TEXT("Accepting GameLift PlayerSession: %s . PlayerId:
 %s"), *playerSessionId, *playerId);
  FString gsId = GetCurrentGameSessionId();
  if (gsId.IsEmpty()) {
    UE_LOG(GameServerLog, Log, TEXT("No GameLift GameSessionId. Returning early!"));
    return false;
  }

  if (!gameLiftSdkModule->AcceptPlayerSession(playerSessionId).IsSuccess()) {
    UE_LOG(GameServerLog, Log, TEXT("PlayerSession not Accepted."));
    return false;
  }

  // Add PlayerSession from internal data structures keeping track of connected players
  connectedPlayerSessionIds.Add(playerSessionId);
  idToPlayerSessionMap.Add(playerSessionId, PlayerSession{ playerId,
 playerSessionId });
  return true;
  #else
  return false;
  #endif
}
```

RemovePlayerSession()

Notifica a Amazon GameLift que un jugador se ha desconectado del proceso del servidor. En respuesta, Amazon GameLift cambia el espacio del jugador para que esté disponible.

Sintaxis

```
FGameLiftGenericOutcome RemovePlayerSession(const FString& playerSessionId)
```

Parámetros

`playerSessionId`

ID único emitido por Amazon GameLift cuando se crea una nueva sesión de jugador.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
bool GameLiftManager::RemovePlayerSession(const FString& playerSessionId)
{
  #if WITH_GAMELIFT
  UE_LOG(GameServerLog, Log, TEXT("Removing GameLift PlayerSession: %s"),
 *playerSessionId);

  if (!gameLiftSdkModule->RemovePlayerSession(playerSessionId).IsSuccess()) {
    UE_LOG(GameServerLog, Log, TEXT("PlayerSession Removal Failed"));
    return false;
  }

  // Remove PlayerSession from internal data structures that are keeping track of
 connected players
  connectedPlayerSessionIds.Remove(playerSessionId);
  idToPlayerSessionMap.Remove(playerSessionId);

  // end the session if there are no more players connected
  if (connectedPlayerSessionIds.Num() == 0) {
    EndSession();
  }

  return true;
  #else
  return false;
  #endif
}
```

DescribePlayerSessions()

Recupera datos de sesión de jugador, incluida la configuración, los metadatos de la sesión y los datos de jugador. Utilice este método para obtener información sobre los siguientes elementos:

- Una sesión para un jugador
- Todas las sesiones del jugador en una sesión de juego
- Todas las sesiones de jugador están asociadas a un único ID de jugador

Sintaxis

```
FGameLiftDescribePlayerSessionsOutcome DescribePlayerSessions(const
 FGameLiftDescribePlayerSessionsRequest &describePlayerSessionsRequest)
```

Parámetros

F GameLiftDescribePlayerSessionsRequest

Un objeto the section called “F GameLiftDescribePlayerSessionsRequest” que describe las sesiones de jugador que recuperar.

Valor devuelto

Si funciona correctamente, devuelve un objeto the section called “F GameLiftDescribePlayerSessionsOutcome” que contiene un conjunto de objetos de sesión de jugador que se ajusta a los parámetros de la solicitud.

Ejemplo

En este ejemplo se solicitan todas las sesiones de jugador conectadas activamente a una sesión de juego específica. Al omitir NextTokeny establecer el valor límite en 10, Amazon GameLift devuelve los registros de las sesiones de los primeros 10 jugadores que coincidan con la solicitud.

```
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()

Envía una solicitud para encontrar nuevos jugadores para ranuras abiertas en una sesión de juego creada con FlexMatch. Para obtener más información, consulta la función FlexMatch de relleno.

Esta acción es asíncrona. Si se emparejan nuevos jugadores, Amazon GameLift proporciona datos actualizados de los emparejadores mediante la función de devolución de llamada. `OnUpdateGameSession()`

Un proceso del servidor solo puede tener una solicitud de reposición de emparejamiento activa a la vez. Para enviar una nueva solicitud, en primer lugar llame a StopMatchBackfill() para cancelar la solicitud original.

Sintaxis

```
FGameLiftStringOutcome StartMatchBackfill (FStartMatchBackfillRequest
 &startBackfillRequest);
```

Parámetros

F StartMatchBackfillRequest

Un StartMatchBackfillRequest objeto que comunica la siguiente información:

- Un ID de ticket que se asignará a la solicitud de reposición. Esta información es opcional; si no se proporciona ningún identificador, Amazon GameLift generará uno.
- El creador de emparejamientos al que se enviará la solicitud. El ARN de configuración completo es obligatorio. Este valor se encuentra en los datos del emparejador de la sesión de juego.
- El ID de la sesión de juego que se va a reponer.

- Datos del emparejador disponibles para los jugadores actuales de la sesión de juego.

Valor devuelto

Devuelve un objeto `StartMatchBackfillOutcome` con el ID del ticket de reposición de emparejamiento o un error con un mensaje de error.

Ejemplo

```
FGameLiftStringOutcome FGameLiftServerSDKModule::StartMatchBackfill(const
 FStartMatchBackfillRequest& request)
{
  #if WITH_GAMELIFT
  Aws::GameLift::Server::Model::StartMatchBackfillRequest sdkRequest;
  sdkRequest.SetTicketId(TCHAR_TO_UTF8(*request.m_ticketId));
  sdkRequest.SetGameSessionArn(TCHAR_TO_UTF8(*request.m_gameSessionArn));

 sdkRequest.SetMatchmakingConfigurationArn(TCHAR_TO_UTF8(*request.m_matchmakingConfigurationArn
  for (auto player : request.m_players) {
    Aws::GameLift::Server::Model::Player sdkPlayer;
    sdkPlayer.SetPlayerId(TCHAR_TO_UTF8(*player.m_playerId));
    sdkPlayer.SetTeam(TCHAR_TO_UTF8(*player.m_team));
    for (auto entry : player.m_latencyInMs) {
      sdkPlayer.WithLatencyMs(TCHAR_TO_UTF8(*entry.Key), entry.Value);
    }

    std::map<std::string, Aws::GameLift::Server::Model::AttributeValue>
 sdkAttributeMap;
    for (auto attributeEntry : player.m_playerAttributes) {
      FAttributeValue value = attributeEntry.Value;
      Aws::GameLift::Server::Model::AttributeValue attribute;
      switch (value.m_type) {
        case FAttributeType::STRING:
          attribute =
 Aws::GameLift::Server::Model::AttributeValue(TCHAR_TO_UTF8(*value.m_S));
        break;
        case FAttributeType::DOUBLE:
          attribute = Aws::GameLift::Server::Model::AttributeValue(value.m_N);
        break;
        case FAttributeType::STRING_LIST:
          attribute =
 Aws::GameLift::Server::Model::AttributeValue::ConstructStringList();
          for (auto sl : value.m_SL) {
```

```
            attribute.AddString(TCHAR_TO_UTF8(*sl));
          };
        break;
        case FAttributeType::STRING_DOUBLE_MAP:
          attribute =
 Aws::GameLift::Server::Model::AttributeValue::ConstructStringDoubleMap();
          for (auto sdm : value.m_SDM) {
            attribute.AddStringAndDouble(TCHAR_TO_UTF8(*sdm.Key), sdm.Value);
          };
        break;
      }
      sdkPlayer.WithPlayerAttribute((TCHAR_TO_UTF8(*attributeEntry.Key)), attribute);
    }
    sdkRequest.AddPlayer(sdkPlayer);
  }
  auto outcome = Aws::GameLift::Server::StartMatchBackfill(sdkRequest);
  if (outcome.IsSuccess()) {
    return FGameLiftStringOutcome(outcome.GetResult().GetTicketId());
  }
  else {
    return FGameLiftStringOutcome(FGameLiftError(outcome.GetError()));
  }
  #else
  return FGameLiftStringOutcome("");
  #endif
}
```

StopMatchBackfill()

Cancela una solicitud de reposición de emparejamiento activa. Para obtener más información, consulta la función de FlexMatch relleno.

Sintaxis

```
FGameLiftGenericOutcome StopMatchBackfill (FStopMatchBackfillRequest
 &stopBackfillRequest);
```

Parámetros

F StopMatchBackfillRequest

Un StopMatchBackfillRequest objeto que identifica el billete de emparejamiento que se va a cancelar:

- ID del ticket que se asignará a la solicitud de reposición
- El emparejador al que se envió la solicitud de reposición
- La sesión de juego asociada a la solicitud de reposición.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

```
FGameLiftGenericOutcome FGameLiftServerSDKModule::StopMatchBackfill(const
 FStopMatchBackfillRequest& request)
{
  #if WITH_GAMELIFT
  Aws::GameLift::Server::Model::StopMatchBackfillRequest sdkRequest;
  sdkRequest.SetTicketId(TCHAR_TO_UTF8(*request.m_ticketId));
  sdkRequest.SetGameSessionArn(TCHAR_TO_UTF8(*request.m_gameSessionArn));

 sdkRequest.SetMatchmakingConfigurationArn(TCHAR_TO_UTF8(*request.m_matchmakingConfigurationArn
  auto outcome = Aws::GameLift::Server::StopMatchBackfill(sdkRequest);
  if (outcome.IsSuccess()) {
    return FGameLiftGenericOutcome(nullptr);
  }
  else {
    return FGameLiftGenericOutcome(FGameLiftError(outcome.GetError()));
  }
  #else
  return FGameLiftGenericOutcome(nullptr);
  #endif
}
```

GetComputeCertificate()

Recupera la ruta al certificado TLS utilizado para cifrar la conexión de red entre el recurso GameLift Anywhere informático de Amazon y Amazon. GameLift Puedes usar la ruta del certificado al registrar tu dispositivo informático en una GameLift Anywhere flota de Amazon. Para obtener más información, consulte, RegisterCompute.

Sintaxis

```
FGameLiftGetComputeCertificateOutcome FGameLiftServerSDKModule::GetComputeCertificate()
```

Valor devuelto

Devuelve un objeto `GetComputeCertificateResponse` que contiene los siguientes elementos:

- CertificatePath: La ruta al certificado TLS de su recurso informático.
- HostName: el nombre de host de su recurso informático.

Ejemplo

```
FGameLiftGetComputeCertificateOutcome FGameLiftServerSDKModule::GetComputeCertificate()
{
  #if WITH_GAMELIFT
  auto outcome = Aws::GameLift::Server::GetComputeCertificate();
  if (outcome.IsSuccess()) {
    auto& outres = outcome.GetResult();
    FGameLiftGetComputeCertificateResult result;
    result.m_certificate_path = UTF8_TO_TCHAR(outres.GetCertificatePath());
    result.m_computeName = UTF8_TO_TCHAR(outres.GetComputeName());
    return FGameLiftGetComputeCertificateOutcome(result);
  }
  else {
    return FGameLiftGetComputeCertificateOutcome(FGameLiftError(outcome.GetError()));
  }
  #else
  return FGameLiftGetComputeCertificateOutcome(FGameLiftGetComputeCertificateResult());
  #endif
}
```

GetFleetRoleCredentials()

Recupera las credenciales del rol de IAM que autorizan GameLift a Amazon a interactuar con otros. Servicios de AWS Para obtener más información, consulte Comunicación con otros recursos de AWS de sus flotas.

Sintaxis

```
FGameLiftGetFleetRoleCredentialsOutcome
 FGameLiftServerSDKModule::GetFleetRoleCredentials(const
 FGameLiftGetFleetRoleCredentialsRequest &request)
```

Parámetros

F GameLiftGetFleetRoleCredentialsRequest

Valor devuelto

Devuelve un objeto the section called “F GameLiftGetFleetRoleCredentialsOutcome”.

Ejemplo

```
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
}
```

Referencia GameLift del SDK (Unreal) del servidor Amazon: tipos de datos

Puedes usar esta referencia del SDK del servidor Amazon GameLift Unreal como ayuda para preparar tu juego multijugador para usarlo con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos y para obtener información sobre el uso del complemento del servidor del SDK de Unreal, consulte Integre Amazon GameLift en un proyecto de Unreal Engine.

Tipos de datos

- 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**
>
> En este tema se describe la API de Amazon GameLift C++ que puede usar al compilar para Unreal Engine. En concreto, esta documentación se aplica al código que se compila con la opción `-DBUILD_FOR_UNREAL=1`.

F ProcessParameters

Este tipo de datos contiene el conjunto de parámetros enviados a Amazon GameLift en unProcessReady().

Propiedades	Descripción
LogParameters	Un objeto con rutas de directorio a los archivos que se generan durante una sesión de juego. Amazon GameLift copia y almacena los archivos para acceder a ellos en el futuro. Tipo: `TArray<FString>` Obligatorio: no
OnHealthCheck	La función de devolución de llamada que Amazon GameLift invoca para solicitar un informe de estado de salud al proceso del servidor. Amazon GameLift llama a esta función cada 60 segundos y espera 60 segundos para recibir una respuesta. El proceso del servidor devuelve `TRUE` si está en buen estado y `FALSE` si no. Si no se devuelve ninguna respuesta, Amazon GameLift registra el proceso del servidor como incorrecto. Esta propiedad es una función de delegació n definida como `DECLARE_DELEGATE_R etVal(bool, FOnHealthCheck) ;` Tipo: `FOnHealthCheck`

	Obligatorio: no
OnProcessTerminate	La función de devolución de llamada que Amazon GameLift invoca para forzar el cierre del proceso del servidor. Tras llamar a esta función, Amazon GameLift espera 5 minutos a que se cierre el proceso del servidor y responde con una ProcessEnding() llamada antes de cerrar el proceso del servidor. Tipo: `FSimpleDelegate` Obligatorio: sí
OnStartGameSession	La función de devolución de llamada que Amazon GameLift invoca para activar una nueva sesión de juego. Amazon GameLift llama a esta función en respuesta a una solicitud de un cliente CreateGameSession. La función callback pasa un GameSessionobjeto, tal y como se define en la Amazon GameLift API Reference. Esta propiedad es una función de delegació n definida como `DECLARE_DELEGATE_O neParam(FOnStartGameSession , Aws::GameLift::Server::Mode l::GameSession);` ; Tipo: `FOnStartGameSession` Obligatorio: sí

OnUpdateGameSession	La función de devolución de llamada que Amazon GameLift invoca para pasar un objeto de sesión de juego actualizado al proceso del servidor. Amazon GameLift llama a esta función cuando se ha procesado una solicitud de relleno de partidas para proporcionar datos actualizados de los emparejadores. Transmite un GameSessionobjeto, una actualización de estado (`updateReason`) y el identificador del ticket de relleno del partido. Esta propiedad es una función de delegació n definida como `DECLARE_DELEGATE_O neParam(FOnUpdateGameSessio n, Aws::GameLift::Server::Mode l::UpdateGameSession);` ; Tipo: `FOnUpdateGameSession` Obligatorio: no
Puerto	El número de puerto en el que escucha el proceso del servidor para recibir conexiones de jugador nuevas. El valor debe estar en el rango de puertos configurado para cualquier flota que implemente esa compilación de servidor de juegos. Este número de puerto se incluye en los objetos de sesión de juego y de jugador, que las sesiones de juego utilizan a la hora de conectarse a un proceso del servidor. Tipo: `int` Obligatorio: sí

UpdateGameSession

Este tipo de datos se actualiza a un objeto de sesión de juego, que incluye el motivo por el que se actualizó la sesión de juego y el ID del ticket de reposición correspondiente si la reposición se utiliza para reponer las sesiones de los jugadores en la sesión de juego.

Propiedades	Descripción
GameSession	Un GameSessionobjeto definido por la GameLift API de Amazon. El objeto `GameSession` contiene propiedades que describen una sesión de juego. Tipo: `Aws::GameLift::Server::GameSession` Obligatorio: no
UpdateReason	El motivo por el que se actualiza la sesión de juego. Tipo: `enum class UpdateReason` • MATCHMAKING_DATA_UPDATED • BACKFILL_FAILED • BACKFILL_TIMED_OUT • BACKFILL_CANCELLED Obligatorio: no
BackfillTicketId	El ID de ticket de reposición que intenta actualizar la sesión de juego. Tipo: `char[]` Obligatorio: no

GameSession

Este tipo de datos proporciona detalles de una sesión de juego.

Propiedades	Descripción
GameSessionId	Un identificador único de la sesión de juego. El ARN de una sesión de juego tiene el siguiente formato: `arn:aws:gamelift:<region>::gamesession/<fleet ID>/<custom ID string or idempotency token>` . Tipo: `char[]` Obligatorio: no
Nombre	Una etiqueta descriptiva de la sesión de juego. Tipo: `char[]` Obligatorio: no
FleetId	Un identificador único para la flota en la que se ejecuta la sesión de juego. Tipo: `char[]` Obligatorio: no
MaximumPlayerSessionCount	El número máximo de conexiones de jugadores a la sesión de juego. Tipo: `int` Obligatorio: no
Puerto	El número de puerto de la sesión de juego. Para conectarse a un servidor de GameLift juegos de Amazon, una aplicación necesita tanto la dirección IP como el número de puerto.

Propiedades	Descripción
	Tipo: `int` Obligatorio: no
IpAddress	La dirección IP de la sesión de juego. Para conectarse a un servidor de GameLift juegos de Amazon, una aplicación necesita tanto la dirección IP como el número de puerto. Tipo: `char[]` Obligatorio: no
GameSessionData	Un conjunto de propiedades de sesión de juego personalizadas, formateadas como un valor de una sola cadena. Tipo: `char[]` Obligatorio: no
MatchmakerData	Información sobre el proceso de emparejam iento que se utilizó para crear la sesión de juego, en sintaxis JSON, con formato como cadena. Además de la configuración de emparejamiento utilizada, contiene datos sobre todos los jugadores asignados al emparejam iento, incluidos los atributos de los jugadores y las asignaciones de los equipos. Tipo: `char[]` Obligatorio: no

Propiedades	Descripción
GameProperties	Un conjunto de propiedades personalizadas de una sesión de juego, con formato como pares clave-valor. Estas propiedades se pasan a una solicitud de iniciar una nueva sesión de juego. Tipo: `GameProperty[]` Obligatorio: no
DnsName	El identificador de DNS asignado a la instancia que ejecuta la sesión de juego. Los valores tienen formato siguiente: • Flotas habilitadas para TLS: `<unique identifier>.<region identifier>.amazongamelift.com` . • Flotas no habilitadas para TLS: `ec2-<unique identifier>.compute.amazonaws.com` . Cuando se conecte a una sesión de juego que se ejecute en una flota habilitada de TLS, debe utilizar el nombre de DNS, no la dirección IP. Tipo: `char[]` Obligatorio: no

F ServerParameters

Información utilizada para mantener la conexión entre un GameLift Anywhere servidor de Amazon y el GameLift servicio de Amazon. Esta información se utiliza al inicializar nuevos procesos de servidor con InitSDK(). Para los servidores alojados en instancias EC2 GameLift gestionadas por Amazon, utilice un objeto vacío.

Propiedades	Descripción
webSocketUrl	`GameLiftServerSdkEndpoint` Amazon GameLift regresa cuando `RegisterCompute` buscas un recurso GameLift Anywhere informático de Amazon. Tipo: `char[]` Obligatorio: sí
processId	Un identificador único registrado en el proceso de servidor que aloja el juego. Tipo: `char[]` Obligatorio: sí
hostId	El `HostID` es el `ComputeName` utilizado cuando registró el recurso informático. Para obtener más información, consulte, RegisterCompute. Tipo: `char[]` Obligatorio: sí
fleetId	El identificador único de la flota en la que está registrado el recurso informático. Para obtener más información, consulte, RegisterCompute. Tipo: `char[]` Obligatorio: sí
authToken	El token de autenticación generado por Amazon GameLift que autentica tu servidor en Amazon GameLift. Para obtener más información, consulte, GetComputeAuthToken.

Propiedades	Descripción
	Tipo: `char[]` Obligatorio: sí

F StartMatchBackfillRequest

Información utilizada para crear una solicitud de reposición de emparejamiento. El servidor del juego comunica esta información a Amazon GameLift en una StartMatchBackfill() llamada.

Propiedades	Descripción
GameSessionArn	Un identificador único de la sesión de juegos. El `GetGameSessionId` de la operación de la API devuelve el identificador en formato de ARN. Tipo: `char[]` Obligatorio: sí
MatchmakingConfigurationArn	Un identificador único, con formato de ARN, que el emparejador utiliza para esta solicitud . El ARN del emparejador para la sesión de juego original se encuentra en el objeto de sesión de juego en la propiedad de datos del emparejador. Puede obtener más información sobre los datos del emparejador en Trabajo con datos del emparejador. Tipo: `char[]` Obligatorio: sí
Players	Un conjunto de datos que representa a todos los jugadores que están en la sesión de juego. El creador de emparejamientos utiliza esta

Propiedades	Descripción
	información para buscar nuevos jugadores que son idóneos para los jugadores actuales. Tipo: `TArray<FPlayer>` Obligatorio: sí
TicketId	Un identificador único para un ticket de solicitud de emparejamiento o reposición de emparejamiento. Si no proporcionas un valor, Amazon GameLift generará uno. Use este identificador para realizar un seguimiento del estado del ticket de reposición de emparejamiento o cancelar la solicitud si es necesario. Tipo: `char[]` Obligatorio: no

FPlayer

Este tipo de datos representa a un jugador en el sistema de emparejamiento. Al iniciar una solicitud de emparejamiento, un jugador tiene un ID de jugador, atributos y, posiblemente, datos de latencia. Amazon GameLift añade la información del equipo después de que se juegue un partido.

Propiedades	Descripción
LatencyInMS	Un conjunto de valores expresados en milisegundos que indican la cantidad de latencia que experimenta un jugador cuando se conecta a una ubicación. Si se utiliza esta propiedad, el jugador solo se empareja con las ubicaciones que aparecen. Si un creador de emparejamientos tiene una regla que evalúa la latencia de los jugadores

Propiedades	Descripción
	, estos deben informar de la latencia para ser emparejados. Tipo: `TMap>FString, int32<` Obligatorio: no
PlayerAttributes	Una colección de pares de clave-valor que contienen información del jugador para su uso en el emparejamiento. Las claves de atributos del jugador deben coincidir con las PlayerAtt ributes utilizadas en un conjunto de reglas de emparejamiento. Para obtener más información sobre los atributos de los jugadores, consulte Attribute Value. Tipo: `TMap>FString, FAttribut eValue<` Obligatorio: no
PlayerId	Un identificador único de un jugador. Tipo: `std::string` Obligatorio: no
Equipo	El nombre del equipo al que está asignado el jugador en un emparejamiento. Defina el nombre del equipo se define en el conjunto de reglas de emparejamiento. Tipo: `FString` Obligatorio: no

F GameLiftDescribePlayerSessionsRequest

Un objeto que especifica las sesiones de jugador que recuperar. El proceso del servidor proporciona esta información con una DescribePlayerSessions() llamada a Amazon GameLift.

Propiedades	Descripción
GameSessionId	Un identificador único de la sesión de juegos. Use este parámetro para solicitar todas las sesiones de jugador de la sesión de juego especificada. El formato de ID de sesión de juego es `FString`. El `GameSessionID` es una cadena de ID personalizada o una Tipo: `std::string` Obligatorio: no
PlayerSessionId	El identificador único de una sesión de jugador. Utilice este parámetro para solicitar una sesión de jugador específica. Tipo: `FString` Obligatorio: no
PlayerId	El identificador único de un jugador. Utilice este parámetro para solicitar todas las sesiones de jugador para un jugador específico. Consulte Generación de ID de jugador. Tipo: `FString` Obligatorio: no
PlayerSessionStatusFilter	El estado de la sesión de jugador para filtrar los resultados. Los posibles estados de sesión de jugador son los siguientes:

Propiedades	Descripción
	• RESERVADO: se ha recibido la solicitud de sesión de jugador, pero el jugador aún no se ha conectado al proceso del servidor o aún no se ha validado. • ACTIVO: el proceso del servidor ha validado el jugador y está conectado. • COMPLETO: la conexión del jugador se ha caído. • TIEMPO DE ESPERA AGOTADO: se ha recibido una solicitud de sesión de jugador, pero el jugador no se ha conectado y/o no se ha validado en el plazo de tiempo de espera (60 segundos). Tipo: `FString` Obligatorio: no
NextToken	El token que indica el inicio de la siguiente página de resultados. Para especificar el inicio del conjunto de resultados, no indique ningún valor. Si se especifica un ID de sesión de jugador, este parámetro se ignora. Tipo: `FString` Obligatorio: no
Límite	El número máximo de resultados que devolver. Si se especifica un ID de sesión de jugador, este parámetro se ignora. Tipo: `int` Obligatorio: no

F StopMatchBackfillRequest

Información utilizada para cancelar una solicitud de reposición de emparejamiento. El servidor del juego comunica esta información al GameLift servicio de Amazon en una StopMatchBackfill() llamada.

Propiedades	Descripción
GameSessionArn	Un identificador único de sesión de juego de la solicitud que se va a cancelar. Tipo: `FString` Obligatorio: sí
MatchmakingConfigurationArn	Un identificador único del emparejador al que se envió esta solicitud. Tipo: `FString` Obligatorio: sí
TicketId	Un identificador único del ticket de solicitud de reposición que se va a cancelar. Tipo: `FString` Obligatorio: sí

F AttributeValue

Utilice estos valores en pares de clave-valor de atributo FPlayer. Este objeto le permite especificar un valor de atributo mediante cualquiera de los tipos de datos válidos: cadena, número, matriz de cadenas o mapa de datos. Cada objeto `AttributeValue` puede utilizar solo una de las propiedades disponibles.

Propiedades	Descripción
attrType	Especifica el tipo de valor del atributo.

Propiedades	Descripción
	Tipo: un valor de enum de `FAttributeType` . Obligatorio: no
S	Representa un valor de atributo de cadena. Tipo: `FString` Obligatorio: no
N	Representa un valor de atributo numérico. Tipo: `double` Obligatorio: no
SL	Representa una matriz de valores de atributos de cadena. Tipo: `TArray<FString>` Obligatorio: no
SDM	Representa un diccionario de claves de cadena y valores dobles. Tipo: `TMap<FString, double>` Obligatorio: no

F GameLiftGetFleetRoleCredentialsRequest

Este tipo de datos proporciona credenciales de rol que amplían el acceso limitado a sus recursos de AWS al servidor de juegos. Para obtener más información, consulte Configurar un rol de servicio de IAM para Amazon GameLift.

Propiedades	Descripción
RoleArn	El nombre de recurso de Amazon (ARN) del rol de servicio que amplía el acceso limitado a sus recursos de AWS. Tipo: `FString` Obligatorio: no
RoleSessionName	El nombre de la sesión que describe el uso de las credenciales del rol. Tipo: `FString` Obligatorio: no

F GameLiftLongOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Resultado	El resultado de la acción. Tipo: `long` Obligatorio: no
ResultWithOwnership	El resultado de la acción, expresado como un valor r, para que el código de llamada pueda tomar posesión del objeto. Tipo: `long&&` Obligatorio: no
Success	Si la acción se realizó correctamente o no. Tipo: `bool`

Propiedades	Descripción
	Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “F GameLiftError” Obligatorio: no

F GameLiftStringOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Resultado	El resultado de la acción. Tipo: `FString` Obligatorio: no
ResultWithOwnership	El resultado de la acción, expresado como un valor r, para que el código de llamada pueda tomar posesión del objeto. Tipo: `FString&&` Obligatorio: no
Success	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “F GameLiftError”

Propiedades	Descripción
	Obligatorio: no

F GameLiftDescribePlayerSessionsOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Resultado	El resultado de la acción. Tipo: the section called “F GameLiftDescribePlayerSessionsResult” Obligatorio: no
ResultWithOwnership	El resultado de la acción, expresado como un valor r, para que el código de llamada pueda tomar posesión del objeto. Tipo: `FGameLiftDescribePlayerSessionsResult&&` Obligatorio: no
Success	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “F GameLiftError” Obligatorio: no

F GameLiftDescribePlayerSessionsResult

Propiedades	Descripción
PlayerSessions	Tipo: `TArray<FGameLiftPlayerSession>` Obligatorio: sí
NextToken	El token que indica el inicio de la siguiente página de resultados. Para especificar el inicio del conjunto de resultados, no indique ningún valor. Si se especifica un ID de sesión de jugador, este parámetro se ignora. Tipo: `FString` Obligatorio: no
Success	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “F GameLiftError” Obligatorio: no

F GenericOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Success	Si la acción se realizó correctamente o no.

Propiedades	Descripción
	Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “F GameLiftError” Obligatorio: no

F GameLiftPlayerSession

Propiedades	Descripción
CreationTime	Tipo: `long` Obligatorio: sí
FleetId	Tipo: `FString` Obligatorio: sí
GameSessionId	Tipo: `FString` Obligatorio: sí
IpAddress	Tipo: `FString` Obligatorio: sí
PlayerData	Tipo: `FString` Obligatorio: sí
PlayerId	Tipo: `FString` Obligatorio: sí

Propiedades	Descripción
PlayerSessionId	Tipo: `FString` Obligatorio: sí
Puerto	Tipo: `int` Obligatorio: sí
Status	Tipo: una enumeración de `PlayerSessionStatus` . Obligatorio: sí
TerminationTime	Tipo: `long` Obligatorio: sí
DnsName	Tipo: `FString` Obligatorio: sí

F GameLiftGetComputeCertificateOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Resultado	El resultado de la acción. Tipo: the section called “F GameLiftGetComputeCertificateResult” Obligatorio: no
ResultWithOwnership	El resultado de la acción, expresado como un valor r, para que el código de llamada pueda tomar posesión del objeto.

Propiedades	Descripción
	Tipo: `FGameLiftGetComputeCertificateResult&&` Obligatorio: no
Success	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “F GameLiftError” Obligatorio: no

F GameLiftGetComputeCertificateResult

La ruta al certificado TLS de su recurso informático y el nombre de host del equipo.

Propiedades	Descripción
CertificatePath	Tipo: `FString` Obligatorio: sí
ComputeName	Tipo: `FString` Obligatorio: sí

F GameLiftGetFleetRoleCredentialsOutcome

Este tipo de datos es el resultado de una acción y produce un objeto con las siguientes propiedades:

Propiedades	Descripción
Resultado	El resultado de la acción. Tipo: the section called “F GetFleetRoleCredentialsResult” Obligatorio: no
ResultWithOwnership	El resultado de la acción, expresado como un valor r, para que el código de llamada pueda tomar posesión del objeto. Tipo: `FGameLiftGetFleetRoleCredentialsResult&&` Obligatorio: no
Success	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “F GameLiftError” Obligatorio: no

F GetFleetRoleCredentialsResult

Propiedades	Descripción
AccessKeyId	El ID de la clave de acceso para autenticar y proporcionar acceso a los recursos de AWS. Tipo: `FString`

Propiedades	Descripción
	Obligatorio: no
AssumedRoleId	El ID del usuario al que pertenece el rol de servicio. Tipo: `FString` Obligatorio: no
AssumedRoleUserArn	El nombre de recurso de Amazon (ARN) del usuario al que pertenece el rol de servicio. Tipo: `FString` Obligatorio: no
Expiration	El tiempo que queda hasta que caduquen las credenciales de la sesión. Tipo: `FDateTime` Obligatorio: no
SecretAccessKey	El ID de clave de acceso secreta para la autenticación. Tipo: `FString` Obligatorio: no
SessionToken	Un token para identificar la sesión activa actual que interactúa con los recursos de AWS. Tipo: `FString` Obligatorio: no

Propiedades	Descripción
Success	Si la acción se realizó correctamente o no. Tipo: `bool` Obligatorio: sí
Error	El error que se genera si la acción no se ha realizado correctamente. Tipo: the section called “GameLiftError” Obligatorio: no

F GameLiftError

Propiedades	Descripción
ErrorType	Tipo de error. Tipo: una enumeración de `GameLiftErrorType` . Obligatorio: no
ErrorName	El nombre del error. Tipo: `std::string` Obligatorio: no
ErrorMessage	Mensaje de error. Tipo: `std::string` Obligatorio: no

Enums

Las enumeraciones definidas para el SDK GameLift del servidor Amazon (Unreal) se definen de la siguiente manera:

F AttributeType

- NONE
- STRING
- DOUBLE
- STRING_LIST
- STRING_DOUBLE_MAP

GameLiftErrorType

Valor de cadena que indica el tipo de error. Los valores válidos son:

- SERVICE_CALL_FAILED: se ha producido un error en la llamada a un servicio de AWS.
- LOCAL_CONNECTION_FAILED — Falló la conexión local con Amazon. GameLift
- NETWORK_NOT_INITIALIZED: la red no se ha inicializado.
- GAMESESSION_ID_NOT_SET: no se ha establecido el ID de sesión de juego.
- BAD_REQUEST_EXCEPTION
- INTERNAL_SERVICE_EXCEPTION
- ALREADY_INITIALIZED: el GameLift servidor o cliente de Amazon ya se inicializó con Initialize ().
- FLEET_MISMATCH: la flota de destino no coincide con la flota de gameSession o playerSession.
- GAMELIFT_CLIENT_NOT_INITIALIZED — El cliente de Amazon no se ha inicializado. GameLift
- GAMELIFT_SERVER_NOT_INITIALIZED — El servidor de Amazon no se ha inicializado. GameLift
- GAME_SESSION_ENDED_FAILED: el SDK de GameLift Amazon Server no pudo contactar con el servicio para informar que la sesión de juego había finalizado.
- GAME_SESSION_NOT_READY: no se activó la sesión de juego del servidor GameLift Amazon.

- GAME_SESSION_READY_FAILED: el SDK de GameLift Amazon Server no pudo contactar con el servicio para informar que la sesión de juego estaba lista.
- INITIALIZATION_MISMATCH: se llamó a un método de cliente después de Server::Initialize() o viceversa.
- NOT_INITIALIZED — El GameLift servidor o cliente de Amazon no se ha inicializado con Initialize ().
- NO_TARGET_ALIASID_SET: no se ha establecido un aliasId de destino.
- NO_TARGET_FLEET_SET: no se ha establecido una flota de destino.
- PROCESS_ENDING_FAILED: el SDK de GameLift Amazon Server no pudo contactar con el servicio para informar que el proceso está finalizando.
- PROCESS_NOT_ACTIVE — El proceso del servidor aún no está activo, no está vinculado a a y no puede aceptarlo ni procesarlo. GameSession PlayerSessions
- PROCESS_NOT_READY: el proceso de servidor aún no está listo para activarse.
- PROCESS_READY_FAILED: el SDK de Amazon GameLift Server no pudo contactar con el servicio para informar que el proceso está listo.
- SDK_VERSION_DETECTION_FAILED: no se pudo detectar la versión del SDK.
- STX_CALL_FAILED: no se pudo realizar una llamada al componente de backend del servidor xSTx.
- STX_INITIALIZATION_FAILED: no se pudo inicializar el componente de backend del servidor xSTx.
- UNEXPECTED_PLAYER_SESSION: el servidor ha detectado una sesión de jugador no registrada.
- WEBSOCKET_CONNECT_FAILURE
- WEBSOCKET_CONNECT_FAILURE_FORBIDDEN
- WEBSOCKET_CONNECT_FAILURE_INVALID_URL
- WEBSOCKET_CONNECT_FAILURE_TIMEOUT
- WEBSOCKET_RETRIABLE_SEND_MESSAGE_FAILURE: error recuperable al enviar un mensaje al servicio. GameLift WebSocket
- WEBSOCKET_SEND_MESSAGE_FAILURE: error al enviar un mensaje al GameLift servicio. WebSocket
- MATCH_BACKFILL_REQUEST_VALIDATION: se ha producido un error en la validación de la solicitud.

- PLAYER_SESSION_REQUEST_VALIDATION: se ha producido un error en la validación de la solicitud.

E PlayerSessionCreationPolicy

Valor de cadena que indica si la sesión de juego acepta jugadores nuevos. Los valores válidos son:

- ACCEPT_ALL: se aceptan todas las sesiones de jugador nuevas.
- DENY_ALL: se rechazan todas las sesiones de jugador nuevas.
- NOT_SET: la sesión de juego no está configurada para aceptar o denegar sesiones de nuevos jugadores.

E PlayerSessionStatus

- ACTIVE
- COMPLETED
- NOT_SET
- RESERVED
- TIMEDOUT

Referencia del SDK del servidor de Unreal Engine de Amazon GameLift 3.x

Puede utilizar esta referencia del SDK del servidor de Unreal Engine de Amazon GameLift 3.x como ayuda para preparar su juego multijugador para su uso con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

Temas

- Referencia del SDK del servidor de Amazon GameLift para Unreal Engine: Acciones
- Referencia del SDK del servidor de Amazon GameLift para Unreal Engine: Tipos de datos

Referencia del SDK del servidor de Amazon GameLift para Unreal Engine: Acciones

Esta referencia del servidor de Amazon GameLift puede ayudarle a preparar los proyectos de juego de Unreal Engine para utilizarlos con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

Esta API se define en `GameLiftServerSDK.h` y `GameLiftServerSDKModels.h`.

Para configurar el complemento Unreal Engine y ver ejemplos de código, consulte Integre Amazon GameLift en un proyecto de Unreal Engine.

- Acciones
- Tipos de datos

AcceptPlayerSession()

Notifica al servicio de Amazon GameLift que un jugador con el ID de sesión de jugador especificado se ha conectado al proceso del servidor y requiere validación. Amazon GameLift verifica que el ID de sesión del jugador es válido, es decir, que el ID de jugador ha reservado una ranura de jugador en la sesión de juego. Una vez completada la validación, Amazon GameLift cambia el estado de la ranura de jugador de RESERVADO a ACTIVO.

Sintaxis

```
FGameLiftGenericOutcome AcceptPlayerSession(const FString& playerSessionId)
```

Parámetros

playerSessionId

ID único emitido por el servicio de Amazon GameLift como respuesta a una llamada a la acción CreatePlayerSession de la API de Amazon GameLift del SDK de AWS. El cliente de juego hace referencia a este ID cuando se conecta al proceso del servidor.

Tipo: FString

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

ActivateGameSession()

Informa al servicio de Amazon GameLift de que el proceso del servidor ha activado una sesión de juego y que está listo para recibir las conexiones de los jugadores. Esta acción debe llamarse como parte de la función de devolución de llamada `onStartGameSession()`, después de completar la inicialización de todas las sesiones de juego.

Sintaxis

```
FGameLiftGenericOutcome ActivateGameSession()
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

DescribePlayerSessions()

Recupera datos de sesión de jugador, incluida la configuración, los metadatos de la sesión y los datos de jugador. Utilice esta acción para obtener información para una única sesión de jugador, para todas las sesiones de jugador de una sesión de juego o para todas las sesiones de jugador asociadas a un solo ID de jugador.

Sintaxis

```
FGameLiftDescribePlayerSessionsOutcome DescribePlayerSessions(const
 FGameLiftDescribePlayerSessionsRequest &describePlayerSessionsRequest)
```

Parámetros

describePlayerSessionsRequest

Es un objeto FDescribePlayerSessionsRequest que describe las sesiones de jugador a recuperar.

Obligatorio: sí

Valor devuelto

Si funciona correctamente, devuelve un objeto FDescribePlayerSessionsRequest que contiene un conjunto de objetos de sesión de jugador que se ajusta a los parámetros de la solicitud. Los objetos de las sesiones de jugador tienen una estructura idéntica al tipo de datos PlayerSession de la API de Amazon GameLift del SDK de AWS.

GetGameSessionId()

Recupera el ID de la sesión de juego alojada actualmente por el proceso del servidor, siempre que esté activo.

Sintaxis

```
FGameLiftStringOutcome GetGameSessionId()
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Si funciona correctamente, devuelve el ID de sesión del juego como objeto `FGameLiftStringOutcome`. Si no funciona, devuelve un mensaje de error.

GetInstanceCertificate()

Recupera la ubicación del archivo de un certificado TLS codificado con PEM que está asociado a la flota y sus instancias. AWS Certificate Manager genera este certificado al crear una nueva flota con la configuración del certificado establecida en GENERADO. Utilice este certificado para establecer una conexión segura con un cliente de juego y para cifrar la comunicación cliente/servidor.

Sintaxis

```
FGameLiftGetInstanceCertificateOutcome GetInstanceCertificate()
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Si se ejecuta correctamente, devuelve un objeto `GetInstanceCertificateOutcome` que contiene la ubicación del archivo de certificado TLS y la cadena de certificados de la flota, que se almacena en la instancia. En la instancia también se almacena un archivo de certificado raíz extraído de la cadena de certificados. Si no funciona, devuelve un mensaje de error.

Para obtener más información sobre el certificado y los datos de la cadena de certificados, consulte Elementos de respuesta de GetCertificate en la referencia de la API de AWS Certificate Manager.

GetSdkVersion()

Devuelve el número de versión actual del SDK integrado en el proceso del servidor.

Sintaxis

```
FGameLiftStringOutcome GetSdkVersion();
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Si funciona correctamente, devuelve la versión del SDK actual como objeto `FGameLiftStringOutcome`. La cadena devuelta solo incluye el número de versión (por ejemplo, («3.1.5»). Si no funciona, devuelve un mensaje de error.

Ejemplo

```
Aws::GameLift::AwsStringOutcome SdkVersionOutcome =
    Aws::GameLift::Server::GetSdkVersion();
```

InitSDK()

Inicializa el SDK de Amazon GameLift. Este método debe llamarse en el momento del lanzamiento, antes de cualquier otra inicialización relacionada con Amazon GameLift.

Sintaxis

```
FGameLiftGenericOutcome InitSDK()
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

ProcessEnding()

Informa al servicio de Amazon GameLift de que el proceso del servidor se va a detener. Este método debe llamarse después de realizar las demás tareas de limpieza, incluido el cierre de todas las

sesiones de juego activas. Se debe salir de este método con un código de salida de 0; un código de salida que no sea 0 provoca un mensaje de evento que afirma que no se ha salido del proceso correctamente.

Sintaxis

```
FGameLiftGenericOutcome ProcessEnding()
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

ProcessReady()

Notifica al servicio de Amazon GameLift que el proceso del servidor está listo para alojar sesiones de juego. Llame a este método después de invocar correctamente a InitSDK() y completar las tareas de configuración necesarias antes de que el proceso del servidor pueda alojar una sesión de juego. Se debe llamar a este método solo una vez por proceso.

Sintaxis

```
FGameLiftGenericOutcome ProcessReady(FProcessParameters &processParameters)
```

Parámetros

FProcessParameters

Es un objeto FProcessParameters que comunica la siguiente información acerca del proceso del servidor:

- Nombres de métodos de devolución de llamada, implementados en el código de servidor de juegos, que el servicio de Amazon GameLift invoca para comunicarse con el proceso del servidor.
- Número de puerto de escucha del servidor de proceso.
- Ruta a cualquier archivo específico de la sesión de juego que desea que Amazon GameLift capture y almacene.

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Ejemplo

Consulte el código de ejemplo en Using the Unreal Engine Plugin.

RemovePlayerSession()

Informa al servicio de Amazon GameLift de que un jugador con el ID de sesión de jugador especificado se ha desconectado del proceso del servidor. Como respuesta, Amazon GameLift cambia el estado de la ranura de jugador a disponible, por lo que se le puede asignar un jugador nuevo.

Sintaxis

```
FGameLiftGenericOutcome RemovePlayerSession(const FString& playerSessionId)
```

Parámetros

playerSessionId

ID único emitido por el servicio de Amazon GameLift como respuesta a una llamada a la acción CreatePlayerSession de la API de Amazon GameLift del SDK de AWS. El cliente de juego hace referencia a este ID cuando se conecta al proceso del servidor.

Tipo: FString

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

StartMatchBackfill()

Envía una solicitud para encontrar nuevos jugadores para ranuras abiertas en una sesión de juego creada con FlexMatch. Consulte también la acción del SDK de AWS StartMatchBackfill(). Con esta acción, un proceso del servidor de juegos que aloja la sesión de juego puede iniciar solicitudes de reposición de emparejamiento. Puede obtener más información sobre la característica de reposición de FlexMatch.

Esta acción es asíncrona. Si se emparejan correctamente nuevos jugadores, el servicio de Amazon GameLift ofrece datos actualizados del emparejador por medio de la función de devolución de llamada `OnUpdateGameSession()`.

Un proceso del servidor solo puede tener una solicitud de reposición de emparejamiento activa a la vez. Para enviar una nueva solicitud, en primer lugar llame a StopMatchBackfill() para cancelar la solicitud original.

Sintaxis

```
FGameLiftStringOutcome StartMatchBackfill (FStartMatchBackfillRequest
 &startBackfillRequest);
```

Parámetros

FStartMatchBackfillRequest

Un objeto FStartMatchBackfillRequest que comunica la siguiente información:

- Un ID de ticket que se asignará a la solicitud de reposición. Esta información es opcional; si no se proporciona ningún ID, Amazon GameLift generará automáticamente uno.
- El creador de emparejamientos al que se enviará la solicitud. El ARN de configuración completo es obligatorio. Este valor se puede obtener de los datos del creador de emparejamientos de la sesión de juego.
- El ID de la sesión de juego que está en fase de reposición.
- Datos de emparejamiento disponibles para los jugadores actuales de la sesión de juego.

Obligatorio: sí

Valor devuelto

Si funciona correctamente, devuelve el ticket de reposición de emparejamiento como un objeto `FGameLiftStringOutcome`. Si no funciona, devuelve un mensaje de error. Podrá realizar el seguimiento del estado del ticket con la acción del SDK de AWS DescribeMatchmaking().

StopMatchBackfill()

Cancela una solicitud de reposición de emparejamiento activa que se creó con StartMatchBackfill(). Consulte también la acción del SDK de AWS StopMatchmaking(). Puede obtener más información sobre la característica de reposición de FlexMatch.

Sintaxis

```
FGameLiftGenericOutcome StopMatchBackfill (FStopMatchBackfillRequest
 &stopBackfillRequest);
```

Parámetros

StopMatchBackfillRequest

Un objeto FStopMatchBackfillRequest que identifica el ticket de emparejamiento que se va a cancelar:

- ID del ticket asignado a la solicitud de reposición que se va a cancelar
- el creador de emparejamientos al que se envió la solicitud de reposición
- sesión de juego asociada a la solicitud de reposición

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

TerminateGameSession()

Este método está obsoleto con la versión 4.0.1. En su lugar, el proceso del servidor debería llamar a ProcessEnding() una vez finalizada la sesión de juego.

Informa al servicio de Amazon GameLift de que el proceso del servidor ha finalizado la sesión de juego actual. Se llama a esta acción cuando el proceso del servidor permanece activo y listo para alojar una nueva sesión de juego. Solo debe llamarse una vez finalizado el procedimiento de finalización de la sesión de juego, ya que indica a Amazon GameLift que el proceso del servidor está disponible inmediatamente para alojar una nueva sesión de juego.

No se llamará a esta acción si el proceso del servidor se interrumpe una vez finalizada la sesión de juego. En su lugar, se llamará a ProcessEnding() para indicar que tanto la sesión de juego como el proceso del servidor están finalizando.

Sintaxis

```
FGameLiftGenericOutcome TerminateGameSession()
```

Parámetros

Esta acción no tiene parámetros.

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

UpdatePlayerSessionCreationPolicy()

Actualiza la capacidad de la sesión de juego actual para aceptar sesiones de jugador nuevas. Una sesión de juego se puede configurar para que acepte o deniegue todas las sesiones nuevas de los jugadores. (Consulte también la acción `UpdateGameSession()` en la Referencia de la API del servicio de Amazon GameLift).

Sintaxis

```
FGameLiftGenericOutcome UpdatePlayerSessionCreationPolicy(EPlayerSessionCreationPolicy
 policy)
```

Parámetros

Política

Valor que indica si la sesión de juego acepta jugadores nuevos.

Tipo: `EPlayerSessionCreationPolicy` enum. Los valores válidos son:

- ACCEPT_ALL: se aceptan todas las sesiones de jugador nuevas.
- DENY_ALL: se rechazan todas las sesiones de jugador nuevas.

Obligatorio: sí

Valor devuelto

Devuelve un resultado genérico correcto o erróneo con un mensaje de error.

Referencia del SDK del servidor de Amazon GameLift para Unreal Engine: Tipos de datos

Esta referencia del servidor de Amazon GameLift puede ayudarle a preparar los proyectos de juego de Unreal Engine para utilizarlos con Amazon GameLift. Para obtener más información sobre el proceso de integración, consulte Adición de Amazon GameLift al servidor de juegos.

Esta API se define en `GameLiftServerSDK.h` y `GameLiftServerSDKModels.h`.

Para configurar el complemento Unreal Engine y ver ejemplos de código, consulte Integre Amazon GameLift en un proyecto de Unreal Engine.

- Acciones
- Tipos de datos

FDescribePlayerSessionsRequest

Este tipo de datos se utiliza para especificar qué sesión o sesiones de jugador recuperar. Puede utilizarlo para las siguientes tareas:

- Proporcionar un PlayerSessionId para solicitar una sesión de jugador específica.
- Proporcionar un GameSessionId para solicitar todas las sesiones de jugador de la sesión de juego especificada.
- Proporcionar un PlayerId para solicitar todas las sesiones de jugador del jugador especificado.

Para grandes conjuntos de sesiones de jugador, utilice los parámetros de paginación para recuperar los resultados en bloques consecutivos.

Contenido

GameSessionId

Identificador único de la sesión de juego. Use este parámetro para solicitar todas las sesiones de jugador de la sesión de juego especificada. El formato de ID de la sesión de juego es el siguiente: `arn:aws:gamelift:<region>::gamesession/fleet-<fleet ID>/<ID string>`. El valor de la <cadena de ID> es una cadena de ID personalizada o (si se especificó una cuando se creó la sesión de juego) una cadena generada.

Tipo: String

Requerido: No

Límite

Número máximo de resultados a devolver. Use este parámetro con NextToken para obtener resultados en un conjunto de páginas secuenciales. Si se especifica un ID de sesión de jugador, este parámetro se ignora.

Tipo: entero

Obligatorio: no

NextToken

Token que indica el inicio de la siguiente página de resultados secuencial. Utilice el token devuelto con una llamada anterior a esta acción. Para especificar el inicio del conjunto de resultados, no indique ningún valor. Si se especifica un ID de sesión de jugador, este parámetro se ignora.

Tipo: String

Requerido: No

PlayerId

Identificador único de un jugador. Los ID de jugador los define el desarrollador. Consulte Generación de ID de jugador.

Tipo: String

Requerido: No

PlayerSessionId

Identificador único de una sesión de jugador.

Tipo: String

Requerido: No

PlayerSessionStatusFilter

Estado de la sesión de juego para filtrar los resultados. Los posibles estados de sesión de jugador son:

- RESERVED: se ha recibido la solicitud de sesión de jugador, pero el jugador aún no se ha conectado al proceso del servidor o aún no se ha validado.
- ACTIVE: el proceso del servidor ha validado el jugador y actualmente está conectado.
- COMPLETED: ha caído la conexión del jugador.
- TIMEDOUT: se ha recibido una solicitud de sesión de jugador, pero el jugador no se ha conectado y/o no se ha validado en el plazo de tiempo de espera (60 segundos).

Tipo: String

Requerido: No

FProcessParameters

Este tipo de datos contiene el conjunto de parámetros enviado a un servicio de Amazon GameLift en una llamada a ProcessReady().

Contenido

puerto

Es el número de puerto al que escucha el proceso del servidor para conexiones de jugador nuevas. El valor debe estar en el rango de puertos configurado para cualquier flota que implemente esa compilación de servidor de juegos. Este número de puerto se incluye en los objetos de sesión de juego y de jugador, que las sesiones de juego utilizan a la hora de conectarse a un proceso del servidor.

Tipo: entero

Obligatorio: sí

logParameters

Objeto con una lista de rutas de directorio a archivos de log de la sesión de juego.

Tipo: TArray<FString>

Obligatorio: no

onStartGameSession

Nombre de la función de devolución de llamada que el servicio de Amazon GameLift invoca para activar una nueva sesión de juego. Amazon GameLift llama a esta función como respuesta a la solicitud de cliente CreateGameSession. La función de devolución de llamada toma un objeto GameSession (definido en la Referencia de la API del servicio de Amazon GameLift).

Tipo: FOnStartGameSession

Obligatorio: sí

onProcessTerminate

Nombre de la función de devolución de llamada que el servicio de Amazon GameLift invoca para forzar el cierre del proceso de servidor. Después de llamar a esta función, Amazon GameLift

espera cinco minutos hasta que el proceso de servidor se cierre y responde con una llamada a ProcessEnding() antes de cerrar el proceso de servidor.

Tipo: FSimpleDelegate

Obligatorio: no

onHealthCheck

Nombre de la función de devolución de llamada que invoca el servicio de Amazon GameLift para solicitar un informe de estado del proceso de servidor. Amazon GameLift llama a esta función cada 60 segundos. Después de llamar a esta función, Amazon GameLift espera una respuesta durante 60 segundos y si no recibe ninguna, registra el proceso del servidor como en mal estado.

Tipo: FOnHealthCheck

Obligatorio: no

onUpdateGameSession

Nombre de la función de devolución de llamada que el servicio de Amazon GameLift invoca para pasar un objeto de sesión de juego actualizado al proceso de servidor. Amazon GameLift llama a esta función cuando se ha procesado una solicitud de reposición de emparejamiento para proporcionar datos actualizados de los emparejadores. Pasa un objeto GameSession, una actualización de estado (`updateReason`) y el ID del ticket de reposición de emparejamiento.

Tipo: FOnUpdateGameSession

Obligatorio: no

FStartMatchBackfillRequest

Este tipo de datos se utiliza para enviar una solicitud de reposición de emparejamiento. La información se comunicará al servicio de Amazon GameLift en una llamada a StartMatchBackfill().

Contenido

GameSessionArn

Identificador único de la sesión de juego. La acción de la API GetGameSessionId() devuelve el identificador en formato de ARN.

Tipo: FString

Obligatorio: sí

MatchmakingConfigurationArn

Identificador único, en forma de un ARN, que el creador de emparejamientos utiliza para esta solicitud. Para encontrar el creador de emparejamientos que se usó para crear la sesión de juego original, busque en el objeto de sesión de juego, en la propiedad de datos del creador de emparejamientos. Puede obtener más información sobre los datos del emparejador en Trabajo con datos del emparejador.

Tipo: FString

Obligatorio: sí

Players

Un conjunto de datos que representa a todos los jugadores que están actualmente en la sesión de juego. El creador de emparejamientos utiliza esta información para buscar nuevos jugadores que son idóneos para los jugadores actuales. Para obtener una descripción del formato del objeto Player, consulte Guía de referencia de la API de Amazon GameLift. Para encontrar los atributos, ID y asignaciones de equipo del jugador, busque en el objeto de sesión de juego, en la propiedad de datos del creador de emparejamientos. Si el creador de emparejamientos utiliza latencia, recopile la latencia actualizada para la región actual e inclúyala en los datos de cada jugador.

Tipo: TArray<FPlayer>

Obligatorio: sí

TicketId

Identificador único para un ticket de solicitud de emparejamiento o reposición de emparejamiento. Si no se proporciona ningún valor aquí, Amazon GameLift generará uno en forma de UUID. Use este identificador para realizar un seguimiento del estado del ticket de reposición de emparejamiento o cancelar la solicitud si es necesario.

Tipo: FString

Obligatorio: no

FStopMatchBackfillRequest

Este tipo de datos se utiliza para cancelar una solicitud de reposición de emparejamiento. La información se comunicará al servicio de Amazon GameLift en una llamada a StopMatchBackfill().

Contenido

GameSessionArn

Identificador único de sesión de juego asociado a la solicitud que se va a cancelar.

Tipo: FString

Obligatorio: sí

MatchmakingConfigurationArn

Identificador único del creador de emparejamientos al que se envió esta solicitud.

Tipo: FString

Obligatorio: sí

TicketId

Identificador único del ticket de solicitud de reposición que se va a cancelar.

Tipo: FString

Obligatorio: sí

Eventos de ubicación de sesión de juego

Amazon GameLift emite eventos para cada solicitud de ubicación de sesión de juego a medida que se procesa. Puede publicar estos eventos en un tema de Amazon SNS, como se describe en Configuración de la notificación de eventos para la ubicación de sesiones de juego.. Estos eventos también se transmiten a Amazon CloudWatch Events prácticamente en tiempo real y con el máximo esfuerzo.

En este tema se describe la estructura de los eventos de ubicación de las sesiones de juego y se proporciona un ejemplo para cada tipo de evento. Para obtener más información sobre el estado de las solicitudes de ubicación de sesiones de juego, consulta GameSessionPlacementla referencia de la GameLift API de Amazon.

Sintaxis de eventos de ubicación

Los eventos se representan como objetos JSON. La estructura de eventos se ajusta al patrón de CloudWatch eventos, con campos de nivel superior similares y detalles específicos del servicio.

Los campos de nivel superior incluyen lo siguiente (consulte el patrón de eventos para obtener más información):

versión

Este campo siempre está establecido en 0 (cero).

id

Un identificador de seguimiento único para el evento.

tipo-detalle

El valor siempre es `GameLift Queue Placement Event`.

source

El valor siempre es `aws.gamelift`.

cuenta

La AWS cuenta que se utiliza para gestionar Amazon GameLift.

hora

La marca temporal del evento.

región

La AWS región en la que se está procesando la solicitud de colocación. Esta es la región en la que se encuentra la cola de sesiones de juego en uso.

resources

Valor ARN de la cola de sesiones de juego que está procesando la solicitud de ubicación.

PlacementFulfilled

La solicitud de ubicación se ha completado satisfactoriamente. Se ha iniciado una nueva sesión de juego y se han creado nuevas sesiones de jugadores para cada uno de los jugadores que figuran en la solicitud de ubicación de la sesión de juego. La información sobre la conexión de los jugadores está disponible.

Sintaxis detallada:

placementId

Un identificador único asignado a la solicitud de ubicación de sesión de juego.

port

El número de puerto de la nueva sesión de juego.

gameSessionArn

Un identificador de ARN de la nueva sesión de juego.

ipAddress

La dirección IP de la sesión de juego.

dnsName

El identificador de DNS asignado a la instancia que ejecuta la nueva sesión de juego. El formato del valor varía en función de si la instancia que ejecuta la sesión de juego está habilitada para TLS. Cuando se conecte a una sesión de juego que se ejecute en una flota habilitada de TLS, debe utilizar el nombre de DNS, no la dirección IP.

Flotas habilitadas para TLS: `<unique identifier>.<region identifier>.amazongamelift.com`.

Flotas no habilitadas para TLS: `ec2-<unique identifier>.compute.amazonaws.com`.

startTime

Marca de tiempo que indica cuándo se puso esta solicitud en la cola.

endTime

Marca de tiempo que indica cuándo se tramitó la solicitud.

gameSessionRegion

AWS Región de la flota que alberga la sesión de juego. Esto corresponde al token de región del`GameSessionArn`.

placedPlayerSessions

La colección de sesiones de jugador que se creó para cada jugador en la solicitud de ubicación de la sesión de juego.

Ejemplo

```
{
  "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 solicitud de colocación se canceló con una llamada al GameLift servicio [StopGameSessionPlacement](#).

Detalles:

placementId

Un identificador único asignado a la solicitud de ubicación de sesión de juego.

startTime

Marca de tiempo que indica cuándo se puso esta solicitud en la cola.

endTime

Marca de tiempo que indica cuándo se canceló la solicitud.

Ejemplo

```
{
  "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

La ubicación de la sesión de juego no se tramitó correctamente antes de que expirara el límite de tiempo de la cola. La solicitud de ubicación se puede volver a enviar según sea necesario.

Detalles:

placementId

Un identificador único asignado a la solicitud de ubicación de sesión de juego.

startTime

Marca de tiempo que indica cuándo se puso esta solicitud en la cola.

endTime

Marca de tiempo que indica cuándo se canceló la solicitud.

Ejemplo

```
{
  "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 no GameLift ha podido tramitar la solicitud de sesión de juego. Por lo general, esto se debe a un error interno inesperado. La solicitud de ubicación se puede volver a enviar según sea necesario.

Detalles:

placementId

Un identificador único asignado a la solicitud de ubicación de sesión de juego.

startTime

Marca de tiempo que indica cuándo se puso esta solicitud en la cola.

endTime

Marca de tiempo que indica cuándo falló la solicitud.

Ejemplo

```
{
  "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"
  }
}
```

Generación de estimaciones de precios de Amazon GameLift

Con AWS Pricing Calculator, puede crear una estimación de precios para Amazon GameLift. No será necesaria una Cuenta de AWS ni disponer de conocimientos avanzados de AWS para utilizar la calculadora.

La calculadora de AWS Pricing Calculator le ayudará a tomar las decisiones en relación con los costos del servicio y le ofrecerá una idea del precio que podría suponer Amazon GameLift en su proyecto de juego. Si aún no está seguro de cómo piensa usar Amazon GameLift, utilice los valores predeterminados para generar una estimación. Si planifica el uso en producción, la calculadora podrá ayudarle a probar posibles escenarios y a generar estimaciones más precisas.

Puede utilizar AWS Pricing Calculator para generar estimaciones de las siguientes opciones de alojamiento de Amazon GameLift:

- Estimación del alojamiento de Amazon GameLift
- Realización de una estimación de FlexMatch de Amazon GameLift independiente

Estimación del alojamiento de Amazon GameLift

Esta opción proporciona una estimación del costo del alojamiento de sus juegos en los servidores administrados de Amazon GameLift, incluidos los costos de uso de la instancia de servidor y de transferencia de datos. El emparejamiento de FlexMatch está incluido en el costo del alojamiento administrado de Amazon GameLift.

Si aloja servidores de juegos en más de una región de AWS o en más de un tipo de instancia, o planea hacerlo, cree una estimación para cada región y tipo de instancia.

Instancias de Amazon GameLift

Esta sección le resultará útil para realizar una estimación del tipo y de la cantidad de recursos informáticos necesarios para alojar las sesiones de juego de sus jugadores. Amazon GameLift utiliza instancias de Amazon Elastic Compute Cloud (Amazon EC2) para administrar servidores de juegos. En Amazon GameLift, podrá implementar una flota de instancias con un tipo de instancia y un sistema operativo específicos. Si dispone de varias flotas o planea tenerlas, cree una estimación para cada una.

Para empezar, abra la página Configurar Amazon GameLift de AWS Pricing Calculator. Añada una descripción, elija una región y, a continuación, seleccione Estimación del alojamiento de Amazon GameLift (instancia + transferencia de datos de salida). En Instancias de Amazon GameLift, complete los siguientes campos:

- Pico de jugadores simultáneos (CCU pico)

  Este es el número máximo de jugadores que pueden conectarse a los servidores de juegos al mismo tiempo. Este campo indica la capacidad de alojamiento que necesita Amazon GameLift para satisfacer la demanda máxima de jugadores. Especifique el número máximo diario de jugadores que espera recibir mediante instancias en la región de AWS elegida.

  Por ejemplo, si desea que se conecten 1000 jugadores a su juego a la vez, mantenga el valor predeterminado de **1000**.

- CCU medio por hora como porcentaje del CCU pico diario

  Este es el número medio de jugadores simultáneos por hora durante un periodo de 24 horas. Utilizamos ese valor para realizar una estimación de la capacidad de alojamiento continua que necesita mantener Amazon GameLift para sus jugadores. Si no sabe con seguridad el valor porcentual que debe utilizar, mantenga el valor predeterminado del **50** por ciento. Para los juegos con una demanda de jugadores estable, le recomendamos que especifique un valor del **70** por ciento.

  Por ejemplo, si su juego tiene un valor de CCU medio por hora de 6000 y un valor CCU pico de 10 000, especifique el valor del **60** por ciento.

- Sesiones de juego por instancia

  Es el número de sesiones de juego que cada una de las instancias del servidor de juegos puede alojar simultáneamente. Los factores que pueden afectar a ese número son los requisitos de los recursos del servidor de juegos, la cantidad de jugadores que alojar en cada sesión de juego y las expectativas de rendimiento de los jugadores. Si para ese juego conoce el número de sesiones simultáneas, especifíquelo. También puede mantener el valor predeterminado de **20**.

- Jugadores por sesión de juego

  Es el número medio de jugadores que se conectan a una sesión de juego, tal y como se define en el diseño del juego. Si dispone de modos de juego con un número diferente de jugadores, realice una estimación del número medio de jugadores por sesión de juego en todo el juego. El valor predeterminado es **8**.

- % de búfer de inactividad de instancia

  Es el porcentaje de capacidad de alojamiento no utilizada que debe mantenerse como reserva para hacer frente a los picos repentinos de demanda de jugadores. El tamaño del búfer es un porcentaje de la cantidad total de instancias de una flota. El valor predeterminado es del **`10`** por ciento.

  Por ejemplo, con un búfer de inactividad del 20 por ciento, una flota que admita a jugadores con 100 instancias activas mantiene 20 instancias inactivas.

- % de instancias de spot

  Las flotas de Amazon GameLift pueden usar una combinación de instancias bajo demanda e instancias de spot. Aunque las instancias bajo demanda ofrecen una disponibilidad más fiable, las instancias de spot ofrecen una alternativa muy rentable. Recomendamos utilizar una combinación de ellas para optimizar tanto el ahorro de costos como la disponibilidad. Para obtener información sobre cómo Amazon GameLift utiliza las instancias de spot, consulte Instancias bajo demanda frente a instancias de spot.

  En este campo, especifique el porcentaje de instancias de spot que mantener en una flota. Recomendamos un porcentaje de instancias de spot de entre el 50 y el 85 por ciento. El valor predeterminado es del **`50`** por ciento.

  Por ejemplo, si implementa una flota con 100 instancias y especifica un **`40`** por ciento, Amazon GameLift trabajará para mantener 60 instancias bajo demanda y 40 instancias de spot.

- Tipo de instancia

  Las flotas de Amazon GameLift pueden utilizar una variedad de tipos de instancias de Amazon EC2 que varían en cuanto a potencia informática, memoria, almacenamiento y capacidades de red. Cuando configure una flota de Amazon GameLift, elija el tipo de instancia que mejor se adapte a las necesidades de su juego. Para obtener información sobre cómo seleccionar un tipo de instancia con Amazon GameLift, consulte Elección de los recursos GameLift informáticos de Amazon.

  Si conoce el tipo de instancia que está utilizando o que piensa utilizar en su flota de Amazon GameLift, elija ese tipo. Si no sabe con seguridad qué tipo elegir, considere la posibilidad de elegir c5.large. Se trata de un tipo de alta disponibilidad con un tamaño y unas capacidades medios.

- Sistema operativo

En este campo se especifica el sistema operativo en el que se ejecutan los servidores de juegos, ya sea Linux o Windows. El valor predeterminado es Linux.

Transferencia de datos salientes (DTO)

En esta sección se le ayudará a realizar una estimación del costo del tráfico entre clientes de juegos y servidores de juegos. Las tarifas por transferencia de datos se aplican únicamente al tráfico saliente. La transferencia de datos entrantes no tiene ningún costo.

En la página Configurar Amazon GameLift de AWS Pricing Calculator, amplíe Transferencia de datos salientes (DTO) y, a continuación, complete los siguientes campos:

- Tipo de estimación de DTO

  Puede realizar una estimación de DTO de cualquiera de las dos formas siguientes, en función de cómo realice el seguimiento de la transferencia de datos del juego.

  - Por mes (en GB): elija este tipo si realiza un seguimiento del tráfico mensual de sus servidores de juegos.
  - Por jugador: elija este tipo si realiza un seguimiento de la transferencia de datos por jugador. Este es el tipo predeterminado.

    En el siguiente campo, se realizará una estimación de DTO por jugador en función del número de horas de juego calculado en la sección anterior.
- DTO al mes (en GB)

  Si eligió el tipo de estimación de DTO Por mes (en GB), especifique el uso mensual estimado de DTO en GB en cada instancia y por región.
- DTO por jugador

  Si ha elegido el tipo de estimación de DTO Por jugador, especifique el uso estimado de DTO por jugador en KB/s. El valor predeterminado es **4**.

Cuando haya terminado de configurar la estimación de precios de Amazon GameLift, elija Agregar a mi estimación. Para obtener más información sobre cómo crear y administrar estimaciones en AWS Pricing Calculator, consulte Creación de una estimación, configuración de un servicio y adición de más servicios en la Guía del usuario de AWS Pricing Calculator.

Realización de una estimación de FlexMatch de Amazon GameLift independiente

Con esta opción se le proporciona una estimación del costo que supone utilizar el emparejamiento de FlexMatch como un servicio independiente mientras aloja sus juegos con otra solución de servidor de juegos. Esto incluye el alojamiento autoadministrado de Amazon GameLift con FleetIQ y el alojamiento en las instalaciones, la red de pares o los tipos de datos primitivos de computación en la nube. Los costos de FlexMatch independiente se basan en la potencia informática utilizada.

Si dispone de más de un emparejador en diferentes regiones de AWS o planea tenerlos, cree una estimación para cada región.

> **Note**
>
> Amazon GameLift FlexMatch está disponible en las siguientes regiones: EE. UU. Este (Norte de Virginia), EE. UU. Oeste (Oregón), Asia Pacífico (Seúl), Asia Pacífico (Sídney), Asia Pacífico (Tokio), Asia Pacífico (Tokio), Europa (Fráncfort) y Europa (Irlanda).

Para empezar, abra la página Configurar Amazon GameLift de AWS Pricing Calculator. Añada una descripción, elija una región y, a continuación, elija Realización de una estimación de FlexMatch de Amazon GameLift independiente. En Amazon GameLift FlexMatch, complete los siguientes campos:

- Pico de jugadores simultáneos (CCU pico)

  Es el número máximo de jugadores que pueden conectarse a los servidores de juegos al mismo tiempo y solicitar el emparejamiento. Especifique el número máximo diario de jugadores que espera emparejar en las sesiones de juego en la región de que elija.

  Por ejemplo, si desea emparejar 1000 jugadores a la vez, mantenga el valor predeterminado de **`1000`**.
- CCU medio por hora como porcentaje del CCU pico diario

  Es la cantidad media de jugadores simultáneos por hora durante un periodo de 24 horas. Este valor ayuda a realizar una estimación del volumen de las solicitudes de emparejamiento. Si no sabe con seguridad el valor porcentual que debe utilizar, mantenga el valor predeterminado del **`50`** por ciento. Para los juegos con una demanda de jugadores estable, le recomendamos que especifique un valor del **`70`** por ciento.

Por ejemplo, si su juego tiene un valor de CCU medio por hora de 6000 y un valor CCU pico de 10 000, especifique el valor del **60** por ciento.

- Número de jugadores por emparejamiento

  Es el número medio de jugadores que se emparejan en una sesión de juego, tal y como se define en el diseño del juego. Si dispone de modos de juego con números diferentes de jugadores, realice una estimación del número medio de jugadores por sesión de juego en todo el juego. El valor predeterminado es **8**.

- Duración del juego (en minutos)

  Es el tiempo medio que los jugadores permanecen en una sesión de juego desde el principio hasta el final. Este valor ayuda a determinar la frecuencia con la que los jugadores pueden necesitar un nuevo emparejamiento. Especifique la duración media del juego en minutos para los jugadores. El valor predeterminado es **1**.

- Complejidad de las reglas de emparejamiento

  La complejidad de las reglas de emparejamiento hace referencia a la cantidad y tipo de reglas utilizadas para emparejar jugadores. El nivel de complejidad del conjunto de reglas ayuda a determinar la cantidad de potencia informática necesaria para cada emparejamiento.

  - Menor complejidad: elija esta opción si su conjunto de reglas de emparejamiento incluye pocas reglas, utiliza tipos de reglas más simples (como las reglas de comparación) y tiene reglas que forman emparejamientos correctos con menos intentos.
  - Mayor complejidad: elija esta opción si su conjunto de reglas de emparejamiento incluye varias reglas, utiliza tipos de reglas más complejos (como reglas de distancia o latencia) y tiene reglas restrictivas que provocan más errores y requieren más intentos de emparejamiento.

  Para obtener más información sobre la complejidad de las reglas y los precios, consulte Amazon GameLift FlexMatch en la página de precios de Amazon GameLift.

Cuando haya terminado de configurar la estimación de precios de Amazon GameLift FlexMatch, elija Agregar a mi estimación. Para obtener más información sobre cómo crear y administrar estimaciones en AWS Pricing Calculator, consulte Creación de una estimación, configuración de un servicio y adición de más servicios en la Guía del usuario de AWS Pricing Calculator.

Cuotas y regiones compatibles

Para conocer las cuotas de servicio de Amazon GameLift de AWS, consulte Cuotas de Amazon GameLift.

Para obtener más información sobre cómo solicitar un aumento de cuota para recursos de AWS, consulte Cuotas de servicio de AWS.

Para obtener una lista de las Regiones de AWS de Amazon GameLift compatibles, consulte Regiones de Amazon GameLift.

Notas de GameLift lanzamiento de Amazon

Las notas GameLift de la versión de Amazon proporcionan detalles sobre las nuevas funciones, actualizaciones y correcciones relacionadas con el servicio.

Versiones del SDK

En las siguientes tablas se enumeran todas las GameLift versiones de Amazon con información sobre las versiones del SDK. No es necesario utilizar SDK comparables para las integraciones de servidor y cliente de juegos. Sin embargo, es posible que las versiones anteriores de un SDK no sean totalmente compatibles con las características más recientes de otro.

Para obtener más información sobre GameLift los SDK de Amazon, consulteSoporte de desarrollo con Amazon GameLift.

Para obtener los GameLift SDK de Amazon más recientes, visita el sitio de descargas de los GameLiftSDK de Amazon.

Versión actual

Realtime

para Unreal

osterior

Versiones anteriores

Lanzamiento del servicio	AWS SDK	SDK del servidor				SDK de cliente de Realti
		Complemento de C# para Unity	C++	Complemento de C++ para Unreal	Go	
14 de diciembre de 2023	1.11.225 o posterior	5.1.0	5.1.1	5.1.0	5.0.0	1.2.0
2023-11-02	1.11.193 o posterior	5.1.0	5.1.1	5.1.0	5.0.0	1.2.0
2023-09-28	1.11.144 o posterior	5.1.0	5.1.1	5.1.0	5.0.0	1.2.0
2023-08-17	1.11.144 o posterior	5.1.0	5.1.1	5.1.0	5.0.0	1.2.0
2023-07-27	1.11.111 o posterior	5.1.0	5.1.0	5.0.2	5.0.0	1.2.0
2023-06-29	1.11.111 o posterior		5.0.4	5.0.2	5.0.0	1.2.0

Lanzamiento del servicio	AWS SDK	SDK del servidor				SDK de cliente de Realti
		Complemento de C# para Unity	C++	Complemento de C++ para Unreal	Go	
15-06-2020	1.11.87 o posterior		5.0.4	5.0.2	5.0.0	1.2.0
2023-05-25	1.11.87 o posterior		5.0.3	5.0.2	5.0.0	1.2.0
2023-04-20	1.11.63 o posterior				5.0.0	1.2.0
2023-04-13	1.10.21 o posterior				5.0.0	1.2.0
2023-02-09	1.10.21 o posterior			3.4.0	5.0.0	1.2.0
2023-01-31	1.10.21 o posterior			3.4.0	5.0.0	1.2.0
2022-12-01	1.10.21 o posterior			3.4.0		1.2.0
2022-08-25	1.9.333 o posterior		3.4.2	3.4.0		1.2.0

Lanzamiento del servicio	AWS SDK	SDK del servidor				SDK de client de Realti
		Complemento de C# para Unity	C++	Complemento de C++ para Unreal	Go	
2021-10-28	1.9.133 o posterior		3.4.2	3.4.0		1.2.0
2021-06-03	1.8.168 o posterior		3.4.2	3.4.0		1.2.0
2021-03-23	1.8.168 o posterior		3.4.1	3.3.3		1.1.0
2021-03-16	1.8.163 o posterior		3.4.1	3.3.3		1.1.0
2021-02-09	1.8.139 o posterior		3.4.1	3.3.3		1.1.0
2020-12-22	1.8.95 o posterior		3.4.1	3.3.3		1.1.0
2020-11-24	1.8.95 o posterior		3.4.1	3.3.2		1.1.0
2020-11-11	1.8.36 o posterior		3.4.1	3.3.2		1.1.0
2020-09-17	1.8.36 o posterior		3.4.1	3.3.2		1.1.0
2020-08-27	1.7.310 o posterior		3.4.0	3.3.1		1.1.0

Lanzamiento del servicio	AWS SDK	SDK del servidor				SDK de cliente de Realti
		Complemento de C# para Unity	C++	Complemento de C++ para Unreal	Go	
16-04-2020	1.7.310 o posterior		3.4.0	3.3.1		1.1.0
2020-04-02	1.7.310 o posterior		3.4.0			1.1.0
2019-12-19	1.7.249 o posterior		3.4.0			1.1.0
14-11-2019	1.7.210 o posterior		3.4.0			1.1.0
24-10-2019	1.7.210 o posterior		3.4.0			1.1.0
03-09-2019	1.7.175 o posterior		3.4.0			1.1.0
09-07-2019	1.7.140 o posterior		3.3.0			1.0.0
25-04-2019	1.7.91 o posterior		3.3.0			1.0.0
07-03-2019	1.7.65 o posterior		3.3.0			
07-02-2019	1.7.45 o posterior		3.3.0			

Lanzamiento del servicio	AWS SDK	SDK del servidor				SDK de cliente de Realti
		Complemento de C# para Unity	C++	Complemento de C++ para Unreal	Go	
14-12-2018	1.6.20 o posterior		3.3.0			
27-09-2018	1.6.20 o posterior		3.2.1			
14-06-2018	1.4.47 o posterior		3.2.1			
10-05-2018	1.4.47 o posterior		3.2.1			
15-02-2018	1.3.58 o posterior		3.2.1			
08-02-2018	1.3.52 o posterior		3.2.0			
2017-09-01	1.1.43 o posterior		3.1.7			
16-08-2017	1.1.31 o posterior		3.1.7			
16-05-2017	1.0.122 o posterior		3.1.5			
11-04-2017	1.0.103 o posterior		3.1.5			

Lanzamiento del servicio	AWS SDK	SDK del servidor				SDK de cliente de Realti
		Complemento de C# para Unity	C++	Complemento de C++ para Unreal	Go	
21-02-2017	1.0.72 o posterior		3.1.5			
18-11-2016	1.0.31 o posterior		3.1.0			
13-10-2016	1.0.17 o posterior		3.1.0			
01-09-2016	0.14.9 o posterior		3.1.0			
04-08-2016	0.12.16 o posterior		3.0.7			

Notas de la versión

Las siguientes notas de la versión están en orden cronológico, con las últimas actualizaciones en primer lugar. Amazon GameLift se lanzó por primera vez en 2016. Para ver las notas de la versión anteriores a las indicadas aquí, consulte los enlaces de fecha de lanzamiento en Versiones del SDK.

24 de abril de 2024: Amazon GameLift lanza flotas de contenedores

Amazon GameLift ofrece ahora una vista previa de las flotas de contenedores, que le ofrece una portabilidad, escalabilidad, tolerancia a fallos y agilidad mejoradas.

En las flotas de contenedores, las instancias de Amazon EC2 alojan uno o más de sus contenedores. Estos contenedores incluyen el servidor de juegos junto con todo lo que necesite, incluidas las

dependencias y las configuraciones. Algunos ejemplos de dependencias son los SDK y los paquetes de software. Tras subir el contenedor a tu Amazon Elastic Container Registry privado, Amazon GameLift rellena tu flota con el contenedor.

Para funcionar en una flota de contenedores, tu servidor de juegos debe ejecutarse en Linux y estar integrado con Server SDK 5.x. En una flota de contenedores, tienes un control preciso de los recursos de alojamiento para poder optimizar el consumo de recursos, como las unidades de CPU y la memoria. También puedes alojar varios servidores de juegos en un contenedor para reducir el uso de recursos.

Con una flota de contenedores, disfrutarás de muchas de las mismas ventajas que ofrecen otros tipos de flotas, como los tipos de instancias bajo demanda, el escalado (automático y manual), las colas y el matchmaking. También obtienes las mismas métricas que otros tipos de flotas, además de algunas nuevas en el caso de los contenedores. Las flotas de contenedores te permiten llegar a los jugadores de todo el mundo en las siguientes ubicaciones y regiones:

- ap-northeast-1
- ap-northeast-2
- ap-southeast-2
- eu-central-1
- eu-west-1
- us-east-1
- us-west-2

Para llegar a más regiones y zonas locales, crea flotas de contenedores con múltiples ubicaciones.

Más información:

- Gestión del alojamiento con GameLift contenedores de Amazon, Guía para GameLift desarrolladores de Amazon
- CreateContainerGroupDefinition, Referencia de la GameLift API de Amazon

13 de febrero de 2024: Amazon GameLift lanza mejoras en los SDK y simplifica la instalación del GameLift complemento de Amazon para Unreal Engine

Versiones actualizadas del SDK:

- Go Server SDK, versión 5.1.0
- SDK de servidor C#, versión 5.1.2
- SDK de servidor C++, versión 5.1.2

Hemos realizado las siguientes mejoras:

- Se mejoró la confiabilidad del SDK al agregar la reconexión automática en caso de interrupción de la red.
- [Ir] Ahora puede llamar `InitSDK()` con o sin los parámetros del servidor. Los servidores de juegos que se ejecutan en flotas EC2 GameLift gestionadas por Amazon leen los parámetros del servidor directamente de las variables de entorno. Los servidores de juegos de GameLift Anywhere las flotas de Amazon deben llamar `InitSDK()` con los parámetros del servidor.

Versiones de complementos actualizadas:

- GameLift Plugin de Amazon para Unreal Engine, versión 1.1.0
- GameLift Plugin de Amazon para Unity, versión 2.1.0
- Complemento SDK de servidor C++ para Unreal, versión 5.1.1
- Complemento SDK de C# Server para Unity, versión 5.1.2

Hemos realizado las siguientes mejoras:

- [GameLift Plugin de Amazon para Unreal Engine] Se actualizaron las instrucciones de instalación y se simplificó el embalaje. Este complemento ahora incluye la última versión del SDK de C++ Server para Unreal.
- Se actualizaron los complementos para que sean compatibles con la última versión del SDK del GameLift servidor.

Más información:

- Integración de juegos con el GameLift complemento de Amazon para Unreal Engine, Guía para GameLift desarrolladores de Amazon
- Descargas de GameLift complementos y SDK de Amazon

14 de diciembre de 2023: Amazon GameLift añade la posibilidad de actualizar las propiedades del juego de las sesiones de juego activas

Ya has podido configurar las propiedades del juego al crear sesiones de juego y buscar propiedades específicas en las sesiones de juego. Ahora también puedes añadir y actualizar estas propiedades en una sesión de juego activa.

Por ejemplo, tus jugadores votan en un mapa en el que quieren jugar. El cliente del juego llama `UpdateGameSession` para modificar un `GameProperty` valor`{"Key": "map", "Value":"jungle"}`. A continuación, el juego implementa el nuevo mapa para los jugadores de la sesión de juego.

Los administradores del juego también pueden recuperar datos útiles de las propiedades del juego mediante esta `SearchGameSessions` operación. Por ejemplo, los administradores pueden enumerar las sesiones de juego que tienen un `Status` valor de `ACTIVE` y esta propiedad del juego:`{"Key": "map", "Value":"desert"}`.

Más información:

- the section called “Añadir Amazon GameLift a un cliente de juegos”, Guía para GameLift desarrolladores de Amazon
- GameProperty, Referencia de la GameLift API de Amazon
- UpdateGameSession, Referencia de la GameLift API de Amazon
- SearchGameSessions, Referencia de la GameLift API de Amazon

21 de noviembre de 2023: Amazon GameLift lanza el soporte para herramientas de infraestructura como código como Terraform y Pulumi impulsadas por AWS Cloud Control API

Ahora puede gestionar toda su pila de GameLift recursos de Amazon mediante las herramientas de infraestructura como código (IaC). Estas herramientas incluyen AWS CloudFormation, y también herramientas de terceros, como Terraform y Pulumi. Con este soporte adicional, ahora puede centrarse en desarrollar su juego y aprovechar DevOps las estrategias para encargarse de la gestión de los recursos, la CI/CD y el despliegue para sus clientes.

Ahora también puedes aprovisionar y configurar todos los tipos de GameLift recursos de Amazon mediante la API de AWS Cloud Control. Puedes seguir trabajando con los recursos mediante las GameLift API de Amazon o las AWS CloudFormation plantillas de Amazon GameLift.

Para obtener más información sobre los GameLift recursos de Amazon disponibles a través de IaC, consulta la referencia del tipo de GameLift recurso de Amazon Referencia del tipo de GameLift recurso de Amazon.

Además, ahora puedes escalar automáticamente tus flotas mediante AWS CloudFormation plantillas o la API de AWS Cloud Control mediante la nueva propiedad Fleet:. `ScalingPolicies`

La API de Cloud Control ofrece a los desarrolladores un conjunto estándar de API para crear, leer, actualizar, eliminar y enumerar recursos (CRUDL) en cientos de AWS servicios y varias herramientas de terceros, como Terraform y Pulumi.

Más información:

- AWS CloudFormation
- AWS API de control en la nube
- AWS Proveedor de CC Terraform
- Pulumi

16 de noviembre de 2023: Amazon GameLift actualiza el complemento independiente para Unity

Versiones del SDK actualizadas: GameLift plugin de Amazon para Unity, versión 2.0.0

El GameLift complemento de Amazon para Unity proporciona herramientas y flujos de trabajo que simplifican los pasos para poner en marcha tu juego de Unity como alojamiento en la nube con Amazon GameLift. Amazon GameLift es un servicio totalmente gestionado que permite a los desarrolladores de juegos gestionar y escalar servidores de juegos dedicados para juegos multijugador basados en sesiones.

Con esta versión, el complemento de Unity se actualiza para utilizar las GameLift funciones más recientes de Amazon, incluida la versión 5.x del SDK del servidor y la compatibilidad con las pruebas locales con Amazon GameLift Anywhere. El complemento es compatible con las versiones Unity 2021.3 LTS y 2022.3 LTS de Unity.

Las principales características del plugin incluyen:

- Flujos de trabajo de interfaz de usuario guiados en el editor de Unity para los siguientes escenarios:

    - Pruebe la integración de sus juegos con Amazon GameLift utilizando su estación de trabajo local como host. Este flujo de trabajo te ayuda a configurar una GameLift Anywhere flota de Amazon para tu máquina local, lanzar instancias de tu servidor y cliente de juegos, solicitar una sesión de juego a través de Amazon GameLift y unirte al juego.
    - Implemente una solución de alojamiento en la nube para su servidor de juegos integrado con EC2 GameLift gestionado por Amazon y AWS recursos de soporte. Este flujo de trabajo le ayuda a configurar su juego para el alojamiento en la nube y ofrece tres escenarios de implementación:
        - Implementa el servidor del juego en una sola flota.
        - Despliega el servidor del juego en un conjunto de flotas de Spot de bajo coste en varias AWS regiones.
        - Despliega el servidor del juego con un FlexMatch emparejador.
- Posibilidad de configurar perfiles de usuario que se vinculen a un usuario de la AWS cuenta y establecer una AWS región predeterminada. Puede mantener varios perfiles para trabajar en diferentes AWS cuentas, usuarios de cuentas y regiones.
- Comodidades especiales que ayudan a agilizar los procesos de GameLift integración e implementación de Amazon, entre las que se incluyen:
    - Cada solución de alojamiento incluye AWS recursos de apoyo, incluido un grupo de usuarios de Amazon Cognito que proporciona identificadores de jugador únicos y validación de jugadores. Las soluciones también incluyen un depósito de Amazon S3 para almacenamiento, notificaciones de eventos de Amazon SNS, AWS Lambda funciones y otros recursos.
    - Para el Anywhere flujo de trabajo, el complemento automatiza la configuración requerida de los parámetros del servidor.
    - Para el flujo de trabajo de Amazon EC2, cada solución de implementación proporciona un servicio de backend de cliente integrado que utiliza funciones de Lambda. El servicio de back-end se encuentra entre el cliente del juego y el GameLift servicio de Amazon y gestiona todas las llamadas directas al GameLift servicio de Amazon.
- Contenido para las pruebas de integración, incluidos los recursos y el código de un sencillo ejemplo de juego multijugador para ilustrar la integración entre el servidor y el cliente del juego.
- Documentación del complemento con una guía de integración detallada y un código de muestra.

Todos los escenarios de implementación, incluidas las flotas de Amazon EC2 Anywhere y las de Amazon EC2, utilizan AWS CloudFormation plantillas para describir e implementar AWS los recursos de la solución de su juego. Estas plantillas se incluyen en la descarga del GameLift plugin de Amazon. Puedes usarlas tal cual o personalizarlas para tu juego.

Más información:

- Guía GameLift del complemento de Amazon para Unity para el SDK de servidor 5.x, Guía para GameLift desarrolladores de Amazon
- Descarga el plugin desde GitHub
- Acerca del GameLift alojamiento de Amazon
- GameLift Foro de Amazon

2 de noviembre de 2023: Amazon GameLift añade soporte para credenciales compartidas

Versiones del SDK actualizadas: AWS SDK 1.11.193

La nueva función de credenciales GameLift compartidas de Amazon permite que las aplicaciones que se despliegan en flotas de EC2 gestionadas interactúen con otros AWS recursos. Esta actualización afecta a las aplicaciones que se agrupan e implementan junto con los archivos binarios de servidores de juegos integrados en la versión 5.x o posterior del SDK del servidor. (Los archivos ejecutables del servidor de juegos ya pueden solicitar credenciales mediante la acción `GetFleetRoleCredentials()` del SDK del servidor 5.x).

Por ejemplo, si quieres implementar la compilación de tu servidor de juegos con un CloudWatch agente de Amazon para recopilar métricas de instancias de EC2 y otros datos, el agente necesita permiso para interactuar con tus CloudWatch recursos. Para ello, primero debes configurar un rol de IAM (AWS Identity and Access Management IAM) con permisos para usar los CloudWatch recursos y, a continuación, configurar una flota con el rol de IAM y las credenciales compartidas habilitados. Cuando Amazon GameLift implementa la compilación del servidor de juegos en cada instancia de EC2, genera un archivo de credenciales compartido y lo almacena en la instancia. Todas las aplicaciones de la instancia pueden utilizar las credenciales compartidas. Amazon actualiza GameLift automáticamente las credenciales temporales a lo largo de la vida de la instancia.

Puede habilitar las credenciales compartidas al crear una flota de EC2 administrada mediante los siguientes métodos:

- En el flujo de trabajo de creación de flotas de GameLift consolas de Amazon.
- Al llamar a la operación de la API del GameLift servicio de Amazon `CreateFleet` mediante el nuevo parámetro`InstanceRoleCredentialsProvider`.

- Al llamar a la operación AWS CLI `aws gamelift create-fleet` con el parámetro`instance-role-credentials-provider`.

Más información:

- Comunícate con otros AWS recursos de tus flotas, Guía para GameLift desarrolladores de Amazon
- CreateFleet InstanceRoleCredentialsProvider, Referencia de la GameLift API de Amazon
- Configurar un rol de servicio de IAM, Amazon GameLift Developer Guide

28 de septiembre de 2023: Amazon GameLift lanza un nuevo complemento independiente para Unreal Engine

Versiones del SDK actualizadas: GameLift plugin de Amazon para Unreal Engine versión 1.0.0

El GameLift complemento de Amazon para Unreal Engine proporciona herramientas y flujos de trabajo que agilizan los pasos para poner en marcha un juego con Amazon GameLift para el alojamiento en la nube. Amazon GameLift es un servicio totalmente gestionado que permite a los desarrolladores de juegos gestionar y escalar servidores de juegos dedicados para juegos multijugador basados en sesiones. El complemento es compatible con las versiones de UE 5.0, 5.1 y 5.2. Entre las características principales se incluyen:

- Los flujos de trabajo de la IU guiados [en el editor Unreal] recorren las siguientes rutas:
  - Pruebe la integración de sus juegos con Amazon GameLift utilizando su estación de trabajo local como host. Este flujo de trabajo te ayuda a configurar una GameLift Anywhere flota de Amazon para tu máquina local, lanzar instancias de tu servidor y cliente de juegos, solicitar una sesión de juego a través de Amazon GameLift y obtener información de conexión para la nueva sesión de juego.
  - Implemente una solución de alojamiento en la nube de Amazon EC2 para su servidor de juegos integrado. Este flujo de trabajo te ayuda a configurar tu juego para el alojamiento en la nube y ofrece tres escenarios de despliegue diferentes: despliega en una sola flota, despliega en un conjunto de flotas puntuales en varias regiones o despliega en un conjunto de flotas con un FlexMatch sistema de emparejamiento. La solución para cada escenario de implementación incluye GameLift los recursos de Amazon y los AWS recursos de soporte.

- Posibilidad de configurar perfiles de usuario que se vinculen a un usuario de la AWS cuenta y definan una AWS región predeterminada. Puede mantener varios perfiles para trabajar en diferentes AWS cuentas, usuarios de cuentas y regiones.
- Comodidades especiales que ayudan a agilizar los procesos de GameLift integración e implementación de Amazon, entre las que se incluyen:
  - Cada solución de alojamiento incluye AWS recursos de apoyo, como un grupo de usuarios básico de Amazon Cognito que proporciona identificadores de jugador únicos, un depósito de Amazon S3 para almacenamiento, notificaciones de eventos de Amazon SNS y funciones. AWS Lambda
  - Para el flujo de trabajo de Anywhere, el complemento automatiza la configuración necesaria de los parámetros del servidor mediante argumentos de línea de comandos.
  - Para el flujo de trabajo de Amazon EC2, cada solución de implementación proporciona un servicio de backend de cliente integrado que utiliza funciones de Lambda. El servicio de backend recibe las solicitudes de los clientes del juego y las transmite al GameLift servicio de Amazon.
- Contenido para realizar pruebas de integración, que incluye un mapa del juego inicial y dos mapas de prueba con planos básicos y elementos de la IU.
- Documentación del complemento con una guía de integración detallada y un código de muestra.

Todos los escenarios de despliegue, incluidos los de Anywhere las flotas de Amazon EC2, utilizan AWS CloudFormation plantillas para describir las soluciones. El complemento utiliza estas plantillas al implementar GameLift los recursos de Amazon para tu juego. Estas plantillas se incluyen en la descarga del GameLift plugin de Amazon y son editables. Puede utilizarlas tal cual o modificarlas para el juego.

Más información:

- Integración de juegos con el GameLift complemento de Amazon para Unreal Engine, Guía para GameLift desarrolladores de Amazon
- Descarga el plugin desde GitHub
- Acerca del GameLift alojamiento de Amazon
- GameLift Foro de Amazon

17 de agosto de 2023: Amazon GameLift ofrece alojamiento de servidores de juegos con procesadores AWS Graviton

Versiones de SDK actualizadas: AWS SDK 1.11.144

Con Amazon GameLift , ahora puedes alojar tus juegos en la nube mediante instancias EC2 con procesadores AWS Graviton. Diseñadas AWS con procesadores basados en ARM64, las instancias Graviton ofrecen la mejor relación precio-rendimiento para las cargas de trabajo en la nube con EC2, con una mejora de hasta un 40% con respecto a las instancias similares basadas en x86. Los procesadores de Graviton3 más recientes ofrecen un rendimiento informático hasta un 25 % superior al de las versiones anteriores.

Con Amazon GameLift, ahora puedes seleccionar entre estas nuevas instancias de la familia AWS Graviton:

- Instancias basadas en Graviton2: c6g, c6gn, r6g, m6g y g5g
- Instancias basadas en Graviton3: c7g, r7g y m7g

Más información:

- AWS Procesador Graviton: conozca las ventajas y los usos prácticos de las instancias EC2 basadas en Graviton.
- Introducción a Graviton: obtenga una visión general de las instancias basadas en Graviton e información sobre cómo se ejecutan las aplicaciones en ellas en función del sistema operativo, los idiomas y los tiempos de ejecución.

> **Note**
>
> Las instancias de Graviton Arm requieren un GameLift servidor Amazon basado en el sistema operativo Linux. Se requiere el SDK de servidor 5.1.1 o posterior para C++ y C#. Se requiere el SDK de servidor 5.1.1 o posterior para continuar. Estas instancias no out-of-the-box admiten la instalación de Mono en Amazon Linux 2023 (AL2023) o Amazon Linux 2 (AL2).

27 de julio de 2023: Amazon GameLift lanza el SDK de servidor 5.1.0 con soporte adicional para el desarrollo de Unity

Versiones del SDK actualizadas: SDK del servidor para C++, C#/Unity y Unreal 5.1.0

La versión más reciente del SDK para GameLift servidores de Amazon incluye actualizaciones para C++, C# y el complemento Unreal, además de un nuevo complemento para usar con el motor de juegos Unity. Los desarrolladores de juegos integran el SDK GameLift del servidor de Amazon en los servidores de juegos que implementan para hospedarse en Amazon GameLift.

La última versión del SDK del servidor contiene las siguientes actualizaciones, que incluyen una serie de solicitudes de los clientes:

- Descarga paquetes de SDK específicos para cada idioma: el sitio de GameLift descargas actualizado de Amazon contiene paquetes de SDK para cada idioma. Puede descargar las versiones actuales o anteriores.
- Nuevo complemento del SDK del servidor de C# para Unity: el nuevo paquete del SDK del servidor para Unity contiene bibliotecas de C# integradas que puede instalar mediante el administrador de paquetes de Unity Editor (consulte la nueva Guía de integración de Unity). Estas bibliotecas incluyen todas las dependencias necesarias. UnityNuGet Puede utilizar este complemento con Unity 2020.3 LTS, 2021.3 LTS y 2022.3 LTS para Windows y Mac OS. Es compatible con los perfiles .NET Framework y .NET Standard de Unity, con .NET Standard 2.1 y .NET 4.x.
- Solución .NET consolidada para C#: el SDK del servidor para C# ahora es compatible con .NET Framework 4.6.2 (actualizado desde la versión 4.6.1) y .NET 6.0 en una sola solución. .NET Standard 2.1 está disponible con las bibliotecas creadas por Unity.
- Actualizaciones del SDK del servidor 5.1.0
  - [C++, C#, Unreal] Ahora puede llamar a `InitSDK()` con o sin los parámetros del servidor. Los servidores de juegos que se ejecutan en flotas EC2 GameLift gestionadas por Amazon leen los parámetros del servidor directamente de las variables de entorno. Los servidores de juegos de GameLift Anywhere las flotas de Amazon deben llamar `InitSDK()` con los parámetros del servidor.
  - Las llamadas al SDK del servidor [C++, C# y Unreal] han mejorado los mensajes de error.
  - [SDK de C++] Para mejorar los tiempos de compilación del SDK del servidor, el indicador de compilación `-DRUN_CLANG_FORMAT` está deshabilitado de forma predeterminada. Puede habilitarlo con `-DRUN_CLANG_FORMAT=1`.

  - [SDK de C++] Al crear las bibliotecas sin las bibliotecas estándar (`-DGAMELIFT_USE_STD=0`), `InitSDK()` deja de utilizar los tipos de datos `std::`.
- Documentación ampliada del SDK del servidor 5.x
  - Se actualizaron las guías de referencia del SDK del servidor para C++, C#/Unity y Unreal, que incluyen una cobertura ampliada de todos los tipos de datos.
    - Referencia del SDK del servidor de Amazon GameLift 5.x para C# y Unity
    - Referencia del SDK del servidor Amazon GameLift 5.x para C++
    - Referencia del SDK del servidor de Unreal Engine de Amazon GameLift 5.x
  - Nuevas versiones de las guías de integración del SDK del servidor 5 para los complementos de Unity y Unreal
    - Integración de Amazon GameLift en un proyecto de Unity
    - Integre Amazon GameLift en un proyecto de Unreal Engine
- Actualizaciones adicionales de la documentación
  - Se revisó la documentación para las operaciones de la API de GameLift servicios de Amazon GetComputeAccessy GetInstanceAccesspara aclarar los procedimientos de acceso remoto en función de la versión GameLift del SDK del servidor de Amazon en uso.
  - Se revisaron las descripciones GameSessionPlacementpara documentar cómo la información de la sesión de juego es transitoria cuando una colocación está en estado «pendiente».

13 de julio de 2023: Amazon GameLift agrega métricas de hardware de flota

Ahora puede realizar un seguimiento de las métricas de rendimiento del hardware de sus flotas de EC2 GameLift gestionadas por Amazon. Las métricas incluyen métricas de instancias de EC2 sobre el uso de la CPU, el volumen de tráfico de la red y la actividad de lectura/escritura del disco. En el caso de Amazon GameLift, estas métricas describen todas las instancias activas en una ubicación de flota. Puedes ver estas métricas de hardware de flota en el CloudWatch panel de control de Amazon en AWS Management Console. También puedes verlos en la GameLift consola de Amazon en los detalles de la flota.

Más información:

- Supervisión de Amazon GameLift con Amazon CloudWatch(Métricas para flotas), Guía para GameLift desarrolladores de Amazon

29 de junio de 2023: Amazon GameLift lanza el soporte para Amazon Linux 2023

Versiones del SDK actualizadas: AWS SDK 1.11.111

GameLift Los clientes de Amazon ahora pueden usar el sistema operativo Amazon Linux 2023 para alojar sus servidores de juegos. El AL2023 ofrece varias mejoras con respecto al AL2, incluida la seguridad. Este sistema operativo está disponible en todas Regiones de AWS las regiones, excepto en las regiones de China.

Los clientes podrán utilizar los sistemas operativos Linux más recientes y seguir recibiendo actualizaciones de seguridad críticas cuando finalice la compatibilidad con Amazon Linux (AL1) en diciembre de 2023. La compatibilidad con Amazon Linux 2 continuará hasta 2025.

Más información:

- Preguntas frecuentes sobre Amazon GameLift Linux Server
- Comparación entre Amazon Linux 2 y Amazon Linux 2023
- Enlaces de referencia GameLift de la API de Amazon:
  - AWS Acción del SDK `CreateBuild`
  - Comando de la CLI `upload-build`
  - Comando de la CLI `create-build`

25 de mayo de 2023: Amazon GameLift FleetiQ añade un filtro para excluir las ubicaciones de las sesiones de juego en las instancias agotadoras

Versiones del SDK actualizadas: SDK 1.11.87 AWS

Si utilizas Amazon GameLift FleetiQ como alojamiento de juegos, ahora puedes evitar que las sesiones de juego se coloquen en instancias que se están agotando actualmente. Las instancias de vaciado están marcadas como cerradas, pero se pueden seleccionar para alojar nuevas sesiones de juego si no hay otros recursos de alojamiento disponibles. Con esta nueva característica, puede excluir por completo el uso de instancias de vaciado.

Utilice esa característica cuando llame a `ClaimGameServer` para buscar los servidores de juegos disponibles. Añada el nuevo parámetro `FilterOption` y establezca los estados de instancia permitidos como ACTIVOS únicamente. En respuesta, Amazon GameLift FleetiQ solo analiza las instancias activas cuando busca y reclama un servidor de juegos disponible.

Más información:

- ClaimGameServeren la referencia de la GameLift API de Amazon
- Cómo funciona FleetIQ en la guía para desarrolladores de Amazon GameLift FleetIQ

16 de mayo de 2023: Amazon GameLift apoya el etiquetado de asignación de costes para las flotas

GameLift Los clientes de Amazon ahora pueden usar etiquetas de asignación de AWS Billing costos para organizar sus costos de alojamiento de juegos. Puede asignar etiquetas de asignación de costes a los recursos individuales de la flota de Amazon GameLift EC2 para realizar un seguimiento de la contribución de sus flotas a los costes generales de alojamiento.

Más información:

- Gestión de los costos de alojamiento de juegos
- Uso de etiquetas de asignación de costos de AWS en la Guía del usuario de AWS Billing

20 de abril de 2023: Amazon GameLift lanza el soporte para Windows Server 2016

Versiones actualizadas del SDK: AWS SDK 1.11.63

GameLift Los clientes de Amazon ahora pueden usar el sistema operativo Windows Server 2016 para alojar sus servidores de juegos. Este sistema operativo está disponible en todos Regiones de AWS. Los clientes pueden utilizar el sistema operativo Windows más reciente y seguir recibiendo actualizaciones de seguridad críticas cuando Microsoft finalice la compatibilidad con Windows Server 2012 en octubre de 2023.

A partir de hoy, los nuevos clientes que necesiten un entorno de tiempo de ejecución de Windows deberán especificar Windows Server 2016 al crear nuevas compilaciones de servidores de juegos para su alojamiento. Los clientes actuales pueden seguir creando nuevas compilaciones y flotas con Windows Server 2012, pero deben completar la migración con Windows Server 2016 antes de la fecha de fin de la compatibilidad de Microsoft, el 10 de octubre de 2023.

Esta actualización incluye los cambios de servicio siguientes:

- Al crear una compilación de servidor de juegos con comandos de Amazon GameLift SDK o CLI, ahora debe configurar el sistema operativo de forma explícita. Ya no hay ningún valor

predeterminado. Para implementar el servidor de juegos en Windows Server 2016, utilice el valor `WINDOWS_2016`.

- Al crear una compilación de servidor de juegos con la GameLift consola Amazon, debes seleccionar un sistema operativo entre los valores disponibles. Si ya es cliente y tiene flotas activas de Windows Server 2012, puede elegir entre dos opciones: `WINDOWS_2012` o `WINDOWS_2016`.

Más información:

- Enlaces de referencia GameLift de la API de Amazon:
  - Comando de la CLI `upload-build`
  - Comando de la CLI `create-build`
  - AWS Acción del SDK `CreateBuild`
- GameLift Preguntas frecuentes de Amazon para Windows 2012

13 de abril de 2023: Amazon GameLift lanza el SDK de servidor 5.x para Unreal

Versiones actualizadas del SDK: SDK del servidor 5.0.0 para Unreal

La última versión del complemento GameLift ligero de Amazon para Unreal Engine ahora se basa en el SDK 5.x GameLift del servidor Amazon. Para empezar a integrar tu entorno de Unreal Engine con Amazon, GameLift consulta los siguientes enlaces.

Más información:

- Integre Amazon GameLift en un proyecto de Unreal Engine
- Adición de Amazon GameLift al servidor de juegos
- Referencia del SDK del servidor Amazon GameLift 5.x para C++

14 de marzo de 2023: Amazon GameLift lanza una nueva experiencia de consola

La nueva GameLift consola de Amazon incluye estas mejoras:

- Navegación mejorada: el nuevo panel de navegación facilita la navegación entre GameLift los recursos de Amazon.

- Página de inicio de Amazon GameLift : la nueva página de destino proporciona enlaces a documentación útil, muestra una descripción general de alto nivel de Amazon GameLift y brinda soporte a través de enlaces a documentación, preguntas frecuentes y AWS re:Post.
- CloudWatch Métricas de Amazon mejoradas: GameLift las métricas de Amazon ahora están disponibles tanto en la GameLift consola de Amazon como en tus CloudWatch paneles de control. Esta actualización también incluye nuevas métricas de rendimiento, utilización y sesiones de jugadores.

Más información:

- Visualización de los datos de juego en la consola
- Administración de recursos de alojamiento de Amazon GameLift.
- Construyendo un casamentero FlexMatch

14 de febrero de 2023: Amazon GameLift ahora admite el cifrado del lado del servidor para los temas de Amazon SNS

El cifrado del servidor (SSE) para temas de SNS cifra los datos confidenciales en reposo. SSE utiliza claves AWS Key Management Service (AWS KMS) para proteger el contenido de sus temas de SNS.

Más información:

- Configuración de la notificación de eventos para la ubicación de sesiones de juego.
- FlexMatcheventos de emparejamiento
- Cifrado en reposo

9 de febrero de 2023: el SDK GameLift del servidor de Amazon es compatible con .NET 6 con C #10

Versiones actualizadas del SDK: SDK del servidor 5.0.0 para .NET 6. No se requieren actualizaciones del SDK.

Si utilizas la plataforma de desarrollo en tiempo real de Unity, continúa usando el Amazon GameLift server SDK 5.0.0 con .NET 4.6. Unity no es compatible con .NET 6.

Más información:

- Descarga la última versión del SDK del GameLift servidor de Amazon en Amazon GameLift para empezar
- Referencia del SDK del servidor de Amazon GameLift 5.x para C# y Unity

31 de enero de 2023: el SDK GameLift del servidor de Amazon es compatible con el lenguaje Go

Versiones actualizadas del SDK: SDK del servidor 5.0.0 para Go

Más información:

- Descarga la última versión del SDK del GameLift servidor de Amazon en Amazon GameLift para empezar
- Referencia del SDK del servidor de Amazon GameLift para Go

1 de diciembre de 2022: Amazon GameLift lanza Amazon GameLift Anywhere y Amazon GameLift Server SDK 5.0

Versiones del SDK actualizadas: AWS SDK 1.10.21, SDK de servidor 5.0.0 para C++ y C#

Amazon GameLift Anywhere utiliza los recursos del servidor de juegos para alojar los servidores de GameLift juegos de Amazon. Puede usar Amazon GameLift Anywhere para integrar sus propios recursos informáticos con los recursos informáticos de EC2 GameLift gestionados por Amazon para distribuir sus servidores de juegos entre varios tipos de procesamiento. También puedes usar Amazon GameLift Anywhere para probar de forma iterativa los servidores de tus juegos sin tener que subir la versión a Amazon GameLift para cada iteración.

Aspectos destacados:

- Nuevos tipos de GameLift Anywhere flota y cómputo de Amazon
- Registro de recursos GameLift Anywhere informáticos de Amazon
- Ciclo de iteración de pruebas mejorado

Amazon GameLift Server SDK 5.0.0 introduce mejoras en el SDK de servidor existente y un nuevo tipo de recurso, el cómputo. Server SDK 5.0.0 es compatible con Amazon GameLift Anywhere y con el uso de tus propios recursos informáticos para el alojamiento de servidores de juegos.

Más información:

- Referencia del SDK del servidor de Amazon GameLift
- Ubicación de la flota
- Elección de los recursos GameLift informáticos de Amazon
- Crea una GameLift Anywhere flota de Amazon

25 de agosto de 2022: Amazon GameLift lanza el soporte para Zonas Locales

Versiones del SDK actualizadas: AWS SDK 1.9.333

Amazon ya GameLift está disponible en ocho Zonas Locales de los Estados Unidos, por lo que puedes desplegar tus flotas más cerca de los jugadores. Puedes usar todas las GameLift funciones gestionadas por Amazon con las Zonas Locales añadiendo las Zonas Locales a tus flotas.

Las Zonas Locales extienden AWS los recursos y los servicios al borde de la nube, cerca de grandes centros poblacionales, industriales y de tecnología de la información (TI). Esto significa que puede implementar aplicaciones que requieren una latencia de milisegundos de un solo dígito más cerca de los usuarios finales o de los centros de datos en las instalaciones.

Más información:

- Zonas locales
- Ubicación de la flota
- Creación de una flota administrada por Amazon GameLift

28 de junio de 2022: Amazon GameLift lanza una nueva experiencia de consola opcional

La nueva GameLift consola de Amazon incluye estas mejoras:

- Navegación mejorada: el nuevo panel de navegación facilita la navegación entre GameLift los recursos de Amazon.

- Página de inicio de Amazon GameLift : la nueva página de destino proporciona enlaces a documentación útil, muestra una descripción general de alto nivel de Amazon GameLift y brinda soporte a través de enlaces a documentación, preguntas frecuentes y AWS re:Post.
- CloudWatch Métricas de Amazon mejoradas: GameLift las métricas de Amazon ahora están disponibles tanto en la GameLift consola de Amazon como en tus CloudWatch paneles de control. Esta actualización también incluye nuevas métricas de rendimiento, utilización y sesiones de jugadores.

Más información:

- Visualización de los datos de juego en la consola
- Administración de recursos de alojamiento de Amazon GameLift.
- Construyendo un casamentero FlexMatch

15 de febrero de 2022: FlexMatch añade una regla compuesta y mejoras adicionales

FlexMatch los usuarios ahora tienen acceso a las siguientes funciones:

- Regla compuesta: se ha añadido compatibilidad con las reglas de emparejamiento compuestas para emparejamientos de 40 o menos jugadores. Ahora puede utilizar declaraciones lógicas para crear una regla compuesta y formar un emparejamiento. Sin una regla compuesta en su conjunto de reglas, para formar un emparejamiento, todas las reglas del conjunto de reglas deben ser verdaderas. Con las reglas compuestas, puede elegir qué reglas aplicar mediante los siguientes operadores lógicos: `and`, `or`, `not` y `xor`.
- Selección de equipos flexible: se actualizaron las expresiones de las propiedades de emparejamiento para permitir la selección de un subconjunto de todos los equipos disponibles.
- Listas de cadenas más largas: se ha aumentado el número máximo de cadenas de 10 a 100 en una lista de cadenas con los valores de los atributos de los jugadores.

Más información:

- Guía GameLift FlexMatch para desarrolladores de Amazon:
  - FlexMatch tipos de reglas
  - FlexMatch expresiones de propiedades
- AttributeValue: SL

28 de octubre de 2021: Amazon GameLift añade compatibilidad con flotas multirregionales en la región de Asia Pacífico (Osaka); Amazon FleetiQ añade compatibilidad con los procesadores GameLift Graviton2 AWS

Versiones del SDK actualizadas: SDK 1.9.133 AWS

Amazon ya GameLift está disponible en la región Asia Pacífico (Osaka). Los desarrolladores de juegos ahora pueden implementar instancias en Osaka utilizando una flota GameLift multirregional.

Ahora puede utilizar los servidores de juegos alojados en Graviton2, respaldados por la arquitectura de procesador basada en ARM, para lograr un mayor rendimiento a un costo menor en comparación con las opciones de computación equivalentes basadas en Intel.

Aspectos destacados:

- Amazon ya GameLift está disponible en la región Asia Pacífico (Osaka).
- Los grupos de servidores de juegos Amazon GameLift FleetiQ ahora se pueden configurar para administrar las familias de instancias c6g, m6g y r6g de Graviton2.

Más información:

- Flota GameLift multirregional de Amazon
- CreateGameServerGroup
- AWS procesador gravitónico

20 de septiembre de 2021: Amazon GameLift lanza un complemento para Unity

La versión 1.0.0 del GameLift plugin de Amazon para Unity contiene bibliotecas y una interfaz de usuario nativa que facilitan el acceso a GameLift los recursos de Amazon y la integración de Amazon GameLift en tu juego de Unity. Puedes usar el GameLift complemento de Amazon para Unity para acceder a GameLift las API de Amazon e implementar AWS CloudFormation plantillas para escenarios de juego comunes. El complemento también incluye un juego de muestra que funciona con los escenarios de muestra. Puedes usar Amazon GameLift Local para ver los mensajes que se transmiten entre el cliente del juego y el servidor del juego para saber cómo interactúa un juego típico con Amazon GameLift.

El complemento para Unity es compatible con Unity 2019.4 LTS y 2020.3 LTS.

Aspectos destacados:

- Cree, ejecute y modifique un juego de ejemplo con diferentes escenarios, o cree el suyo propio.
- Despliega AWS CloudFormation escenarios de ejemplo para escenarios de juego típicos, como solo autenticación, flotas de una sola región, flotas multirregionales con cola y emparejador personalizado, flotas puntuales con cola y emparejador personalizado, y. FlexMatch

Más información:

- Integración de juegos con el GameLift complemento Amazon para Unity

30 de junio de 2021: FlexMatch agrega la regla BatchDistance

Puede utilizar el tipo de regla batchDistance para especificar un atributo numérico o de cadena, lo que aporta una serie de ventajas a cada segmento.

Aspectos destacados:

- En emparejamientos de gran tamaño (más de 40 jugadores), en lugar de equilibrar uniformemente a los jugadores solo por habilidad, ahora puede conseguir el mismo equilibrio en función de la habilidad, los modos y los mapas. Asegúrese de que todos los jugadores del emparejamiento estén en un grupo de habilidades, agrupe varios atributos numéricos, como la liga o el estilo de juego, y agrupe según cadenas de atributos, como el mapa o el modo de juego. También puede crear expansiones a lo largo del tiempo. Por ejemplo, puede crear una expansión para permitir que un mayor nivel de habilidad entre en el emparejamiento cuanto más tiempo espere el jugador.

  Para emparejamientos de menos de 40 jugadores, puede utilizar una nueva expresión de reglas simplificada.

3 de junio de 2021: actualizaciones del SDK para clientes y servidores en GameLift tiempo real de Amazon

Versiones actualizadas del SDK: SDK del cliente de Realtime 1.2.0, SDK del servidor 3.4.0 para Unreal

Con esta última actualización del SDK, ahora puede integrar IL2CPP en sus aplicaciones móviles que utilizan el SDK del cliente de RTS y seguir las prácticas recomendadas con los marcos. Ahora también puedes compilar el SDK de Amazon GameLift Server para Unreal, versión 4.26. Esta

actualización contiene componentes que se integran con tu servidor de juegos de Windows o Linux, incluidas las versiones C++ y C# del SDK de Amazon GameLift Server, Amazon GameLift Local y un complemento de Unreal Engine.

Aspectos destacados:

- Se añadió soporte para IL2CPP en el SDK del cliente de RTS y para crear las bibliotecas nativas como marcos, de modo que pueda crear clientes de RTS para los dispositivos móviles más recientes.
- Puede utilizar DescribePlayerSessions() para obtener información para una única sesión de jugador, para todas las sesiones de jugador de una sesión de juego o para todas las sesiones de jugador asociadas a un solo ID de jugador.
- Puede usar GetInstanceCertificate() para recuperar la ubicación del archivo de un certificado TLS codificado en PEM que esté asociado a la flota y sus instancias.
- Se creó la compatibilidad con el SDK del servidor para la versión 4.26 de Unreal.
- Se ha comprobado que el SDK de C# existente, versión 4.0.2, es compatible con Unity 2020.3. No se requieren actualizaciones del SDK.

Más información:

- Guía GameLift para desarrolladores de Amazon:
  - DescribePlayerSessions()
  - GetInstanceCertificate()

23 de marzo de 2021: Amazon GameLift añade notificaciones a la ubicación de las sesiones de juego

Versiones del SDK actualizadas: AWS SDK 1.8.168

Ahora puede utilizar los eventos para supervisar la actividad de ubicación de las sesiones de juego para una cola de sesiones de juego. Crea un tema del Amazon Simple Notification Service (Amazon SNS) para publicar notificaciones de eventos o configura el seguimiento de eventos mediante Events. CloudWatch

Aspectos destacados:

- Para cada cola, puede configurar una cadena de texto personalizada para incluirla en todos los mensajes de eventos.
- Al utilizar un tema de Amazon SNS, podrá establecer condiciones de acceso adicionales que limiten la publicación a colas específicas.

Más información:

- Guía GameLift para desarrolladores de Amazon:
  - Configuración de la notificación de eventos para la ubicación de sesiones de juego. (nuevo)
  - Eventos de ubicación de sesión de juego (nuevo)
- Referencia de la API (SDK de AWS)
  - Nuevos parámetros de cola de sesiones de juego `NotificationTarget` y`CustomEventData`: GameSessionQueue,, CreateGameSessionQueueUpdateGameSessionQueue
- GameLiftForo de Amazon

16 de marzo de 2021: Amazon GameLift añade flotas multirregionales y seis nuevas regiones

Versiones del SDK actualizadas: AWS SDK 1.8.163

El alojamiento GameLift gestionado por Amazon ya está disponible en 21 AWS regiones. Las nuevas regiones son Ciudad del Cabo (`af-south-1`), Bahréin (`me-south-1`), Hong Kong (`ap-east-1`), Milán (`eu-south-1`), París (`eu-west-3`) y Estocolmo (`eu-north-1`).

Con la nueva función de flotas GameLift multiubicación de Amazon, ahora puedes configurar una sola flota para alojar tus servidores de juegos en cualquiera de las 20 regiones GameLift compatibles con Amazon o en todas ellas (excepto la región de Pekín). Esta función tiene como objetivo reducir significativamente el trabajo necesario para configurar y mantener los recursos de GameLift alojamiento de Amazon en todo el mundo. Se pueden crear flotas con múltiples ubicaciones en las siguientes AWS regiones: `us-east-1` (Virginia del Norte), `us-west-2` (Oregón), `eu-central-1` (Fráncfort), `eu-west-1` (Irlanda), `ap-southeast-2` (Sídney), `ap-northeast-1` (Tokio) y `ap-northeast-2` (Seúl). En todas las demás regiones, puede seguir configurando flotas de una sola ubicación según sea necesario. Todas las flotas que se crearon antes de esta versión son flotas de una sola ubicación. El uso de flotas con varias ubicaciones no afecta a los costos de alojamiento. GameLiftLos precios de Amazon se basan en el tipo, la ubicación y el volumen de

instancias que utilices. (Para obtener más información, consulta los GameLiftprecios de Amazon). AWS CloudFormation El soporte para flotas con múltiples ubicaciones estará disponible pronto.

> **Note**
>
> Las flotas de varias ubicaciones no están disponibles en las regiones de China. GameLiftLos recursos de Amazon que residen en las regiones de China no pueden interactuar con los recursos de otras GameLift regiones de Amazon ni ser utilizados por ellos.

Aspectos destacados:

- En el caso de una flota con varias ubicaciones, añada explícitamente una lista de ubicaciones remotas. Amazon GameLift despliega instancias del mismo tipo y configuración, incluidas la configuración de compilación y tiempo de ejecución, en la región de origen de la flota y en todas las ubicaciones añadidas.
- Ajuste la configuración de capacidad y el escalado de cada ubicación de forma independiente. Las políticas de escalado automático se aplican a toda la flota, pero puede activarlas o desactivarlas por ubicación.
- Inicie nuevas sesiones de juego en ubicaciones específicas de la flota. Al utilizar las colas de las sesiones de juego o el emparejamiento para ubicar las sesiones de juego, ahora puede priorizar el lugar de inicio de las nuevas sesiones de juego en función de la ubicación, el costo del alojamiento y la latencia de los jugadores.
- Obtén estadísticas de alojamiento en la GameLift consola de Amazon, agregadas para todas las ubicaciones de una flota o desglosadas por cada ubicación de la flota.

Más información:

- Blog de tecnología de juegos de Amazon
- Referencia de la API (SDK de AWS)
  - Nuevas operaciones de ubicación de la flota: CreateFleetLocationsDescribeFleetLocationAttributes, DescribeFleetLocationCapacity,, DescribeFleetLocationUtilization, DeleteFleetLocations
  - Operaciones de flota actualizadas, con un nuevo soporte para múltiples ubicaciones: CreateFleet,, DescribeEC2 UpdateFleetCapacityInstanceLimits,,, DescribeInstancesStopFleetActionsStartFleetActions

  - Se actualizaron las operaciones de ubicación de las sesiones de juego, con una nueva prioridad y capacidad de filtrado:,, CreateGameSessionQueueDescribeGameSessionQueuesUpdateGameSessionQueue
  - Operaciones de creación de sesiones de juego actualizadas, con un nuevo soporte de ubicación: CreateGameSession, DescribeGameSessions, DescribeGameSessionDetails, SearchGameSessions
- Guía GameLift para desarrolladores de Amazon:
  - Ubicaciones GameLift de alojamiento de Amazon (actualizado)
  - Guía de diseño de flotas de Amazon GameLift (nuevo)

    Escalación de la capacidad de alojamiento de Amazon GameLift (actualizado)
  - Diseño de colas de sesiones de juego (nuevo)
  - Visualización de los detalles de la flota (actualizado)
- GameLiftForo de Amazon

9 de febrero de 2021: Amazon GameLift amplía el soporte para instancias AMD independientes FlexMatch

Versiones del SDK actualizadas: AWS SDK 1.8.139

Esta versión incluye las siguientes actualizaciones:

- Los grupos de servidores de juegos Amazon GameLift FleetiQ ahora se pueden configurar para administrar las familias de instancias AMD C5a, M5a y R5a. Los tipos de instancias de Amazon EC2 compatibles, tal y como se indican en la lista GameServerGroup InstanceDefinition, incluyen ahora los siguientes:
  - c5a.large, c5a.xlarge, c5a.2xlarge, c5a.4xlarge, c5a.8xlarge, c5a.12xlarge, c5a.16xlarge y c5a.24xlarge
  - m5a.large, m5a.xlarge, m5a.2xlarge, m5a.4xlarge, m5a.8xlarge, m5a.12xlarge, m5a.16xlarge y m5a.24xlarge
  - r5a.large, r5a.xlarge, r5a.2xlarge, r5a.4xlarge, r5a.8xlarge, r5a.12xlarge, r5a.16xlarge y r5a.24xlarge

  Nota: Actualmente, las instancias AMD para FleetiQ no están disponibles para su uso en la región de China (Pekín). AWS Consulte Disponibilidad de la característica y diferencias en la implementación en China.

- El alojamiento de juegos GameLift gestionado por Amazon ahora es compatible con las instancias de AMD en la región de China (Pekín), gestionadas por Sinnet. Las nuevas familias de instancias de AMD incluyen M5a y R5a. Los tipos de instancias EC2 compatibles, tal y como se indican para Fleet InstanceType, incluyen ahora los siguientes:
  - m5a.large, m5a.xlarge, m5a.2xlarge, m5a.4xlarge, m5a.8xlarge, m5a.12xlarge, m5a.16xlarge y m5a.24xlarge
  - r5a.large, r5a.xlarge, r5a.2xlarge, r5a.4xlarge, r5a.8xlarge, r5a.12xlarge, r5a.16xlarge y r5a.24xlarge
- Amazon ahora se GameLift FlexMatch puede utilizar como una solución de búsqueda de pareja independiente en la región de China (Beijing), operada por Sinnet. Los clientes pueden crear un FlexMatch emparejador en la región de Pekín y configurar el parámetro en STANDALONE. FlexMatchMode Para obtener más información sobre FlexMatch el alojamiento GameLift gestionado por Amazon o con una solución de alojamiento que no sea de Amazon GameLift, consulta la Guía para GameLift FlexMatch desarrolladores de Amazon.
- Al configurar las notificaciones de eventos para Amazon GameLift FlexMatch, ahora puede designar un tema FIFO de Amazon SNS como destino de la notificación. Para obtener más información, consulte:
  - MatchmakingConfiguration NotificationTarget, Referencia de la GameLift API de Amazon
  - Configurar la notificación de FlexMatch eventos, Guía para GameLift FlexMatch desarrolladores de Amazon
  - Presentamos la mensajería Amazon SNS FIFO — F irst-in-first-out Pub/Sub, blog de noticias AWS

22 de diciembre de 2020: el SDK GameLift del servidor de Amazon es compatible con Unreal Engine 4.25 y Unity 2020

Versiones de SDK actualizadas: Amazon GameLift Server SDK 4.0.2, complemento Unreal versión 3.3.3

La última versión del SDK de Amazon GameLift Server contiene los siguientes componentes:

- El complemento de Unreal se ha actualizado para que sea compatible con Unreal Engine 4.25. La API no se ha modificado.
- Se ha comprobado que el SDK de C# existente, versión 4.0.2, es compatible con Unity 2020. No se requieren actualizaciones del SDK.

Descarga la última versión del SDK de Amazon GameLift Server en Amazon GameLift Getting Starting.

24 de noviembre de 2020: Amazon GameLift FlexMatch ya está disponible para juegos alojados en cualquier lugar

Versiones del SDK actualizadas: AWS SDK 1.8.95

Amazon GameLift FlexMatch es un servicio de emparejamiento personalizable para juegos multijugador. Diseñado inicialmente para los usuarios del alojamiento GameLift gestionado por Amazon, ahora se FlexMatch puede integrar en juegos que utilizan otros sistemas de alojamiento peer-to-peer, incluidos la computación local propietaria y los tipos primitivos de computación en la nube. Los juegos que utilizan Amazon GameLift FleetiQ para el alojamiento de juegos en Amazon EC2 ahora pueden implementar el emparejamiento con. FlexMatch

FlexMatch proporciona un sólido algoritmo de emparejamiento y un lenguaje de reglas que le permiten personalizar el proceso de emparejamiento de manera que los jugadores se agrupen en función de sus características clave y de la latencia reportada. Además, FlexMatch ofrece un flujo de trabajo de solicitudes de emparejamiento que admite funciones como los grupos de jugadores, la aceptación de los jugadores y el relleno de partidas. Cuando lo utilizas FlexMatch con el alojamiento GameLift gestionado por Amazon o con Realtime Servers, el emparejador utiliza Amazon automáticamente GameLift para encontrar recursos de alojamiento e iniciar una nueva sesión de juego para las partidas recién formadas. Cuando se usa FlexMatch como un servicio independiente, el emparejador envía los resultados de las partidas a tu juego, que luego puede iniciar una nueva sesión de juego con tu solución de alojamiento.

Las operaciones de la API para FlexMatch forman parte de la API del GameLift servicio de Amazon, que se incluye en el AWS SDK y en AWS Command Line Interface (AWS CLI). Esta versión incluye estas actualizaciones para admitir el emparejamiento independiente:

- El elemento `MatchmakingConfiguration` del recurso de la API tiene los siguientes cambios:
  - Nueva propiedad, `FlexMatchMode` indica si el emparejador se está utilizando con el alojamiento GameLift administrado por Amazon o como emparejamiento independiente.
  - La propiedad `GameSessionQueueArns` no es necesaria cuando `FlexMatchMode` se establece en independiente.
  - Estas propiedades no se utilizan con el emparejamiento independiente: `AdditionalPlayerCount`, `BackfillMode`, `GameProperties` y `GameSessionData`.

- La característica de reposición automática no está disponible con el emparejamiento independiente.

24 de noviembre de 2020: las instancias de AMD ya están disponibles en Amazon GameLift

Versiones del SDK actualizadas: AWS SDK 1.8.95

La lista de tipos de instancias de Amazon EC2 compatibles con Amazon GameLift ahora incluye tres nuevas familias de instancias: C5a, M5a y R5a. Estas familias se componen de instancias optimizadas para la computación de AMD que funcionan con procesadores EPYC de AMD que funcionan con frecuencias de hasta 3,3. GHz. Las instancias de AMD son compatibles con x86; los juegos que se ejecutan actualmente en Amazon se GameLift pueden implementar en los tipos de instancias de AMD sin modificarlos. Las nuevas instancias están disponibles en las siguientes AWS regiones: EE.UU. Este (Norte de Virginia y Ohio), EE.UU. Oeste (Oregón y Norte de California), Centro de Canadá (Montreal), Sudamérica (São Paulo), UE Central (Fráncfort), UE Oeste (Londres e Irlanda), Asia Pacífico Sur (Bombay), Asia Pacífico Noreste (Seúl y Tokio) y Asia Pacífico Sudeste (Singapur y Sídney).

Las nuevas instancias de AMD incluyen:

- c5a.large, c5a.xlarge, c5a.2xlarge, c5a.4xlarge, c5a.8xlarge, c5a.12xlarge, c5a.16xlarge y c5a.24xlarge
- m5a.large, m5a.xlarge, m5a.2xlarge, m5a.4xlarge, m5a.8xlarge, m5a.12xlarge, m5a.16xlarge y m5a.24xlarge
- r5a.large, r5a.xlarge, r5a.2xlarge, r5a.4xlarge, r5a.8xlarge, r5a.12xlarge, r5a.16xlarge y r5a.24xlarge

Más información:

- Blog de tecnología de juegos de Amazon
- Precios de las GameLift instancias de Amazon
- Instancias de Amazon EC2 con procesadores EPYC de AMD
- GameLiftForo de Amazon

11 de noviembre de 2020: actualización de la versión GameLift del SDK del servidor Amazon

Versiones de SDK actualizadas: Amazon GameLift Server SDK 4.0.2

En la nueva versión 4.0.2 del SDK del servidor se ha corregido un problema conocido relacionado con `StartMatchBackfill()` en relación con el funcionamiento de la API. Esta operación ahora devuelve una respuesta correcta a una solicitud de reposición de emparejamientos.

El problema no afectó al proceso de reposición de emparejamientos y no se ha producido ningún cambio en el funcionamiento de esta característica. Es posible que el problema haya afectado a los mensajes de registro y a la gestión de errores en las solicitudes de reposición de emparejamientos.

Descarga la última versión del SDK de Amazon GameLift Server en Amazon GameLift Getting Starting.

5 de noviembre de 2020: nuevas personalizaciones de FlexMatch algoritmos

FlexMatch los usuarios ahora pueden ajustar los siguientes comportamientos predeterminados para el proceso de emparejamiento. Estas personalizaciones se establecen en un conjunto de reglas de emparejamiento. No hay cambios en los GameLift SDK de Amazon.

- Priorice los tickets de reposición: cuando busque emparejamientos aceptables, podrá elegir subir o bajar la prioridad de los tickets de reposición para los emparejamientos. Priorizar los tickets de reposición resulta útil cuando la característica de reposición automática está habilitada. Utilice la propiedad del algoritmo `backfillPriority`.
- Clasifique previamente para optimizar la coherencia y la eficiencia de los emparejamientos: configure el emparejador para que clasifique previamente el grupo de tickets antes de agruparlos por lotes para su evaluación. Al clasificar previamente los tickets en función de los atributos clave de los jugadores, los emparejamientos resultantes suelen tener jugadores más parecidos en esos atributos. También puede aumentar la eficiencia del proceso de evaluación clasificando previamente los mismos atributos que se utilizan en las reglas de emparejamientos. Utilice la propiedad del algoritmo `sortByAttributes` con la propiedad `strategy` establecida en «ordenado».
- Ajuste la forma en que se activan los tiempos de espera de las expansiones: elija entre activar las expansiones en función de la antigüedad del ticket más nuevo (predeterminado) o el más antiguo en un emparejamiento incompleto. Si se activa con el ticket más antiguo, los emparejamientos se completan más rápido, mientras que si se activa con el ticket más nuevo, la calidad del emparejamiento es superior. Utilice la propiedad del algoritmo `expansionAgeSelection`.

17 de septiembre de 2020: Amazon GameLift actualiza el SDK del servidor

Versiones de SDK actualizadas: Amazon GameLift Server SDK 4.0.1

El nuevo SDK del servidor contiene las siguientes actualizaciones:

- Versión de la API de C# 4.0.1
  - La operación de la API TerminateGameSession() ya no es compatible. Sustitúyala por una llamada a ProcessEnding() para finalizar la sesión de juego y el proceso del servidor.
  - Se ha solucionado un problema conocido relacionado con la operación GetInstanceCertificate().
  - La operación GetTerminationTime() ahora devuelve un valor del tipo AwsDateTimeOutcome de datos.
- Versión de la API de C++ 3.4.1
  - La operación TerminateGameSession() ya no es compatible. Sustitúyala por una llamada a ProcessEnding() para finalizar la sesión de juego y el proceso del servidor.
- Versión del complemento de Unreal Engine 3.3.2
  - La operación TerminateGameSession() ya no es compatible. Sustitúyala por una llamada a ProcessEnding() para finalizar la sesión de juego y el proceso del servidor.
  - Se añade la operación de devolución de llamada `OnUpdateGameSession` a FProcessParameters para facilitar la reposición de emparejamientos.

Descarga la última versión del SDK de Amazon GameLift Server en Amazon GameLift Getting Starting.

27 de agosto de 2020: Amazon GameLift FleetiQ para alojamiento de juegos con Amazon EC2 (disponibilidad general)

Versiones del SDK actualizadas: SDK 1.8.36 AWS

La solución Amazon GameLift FleetiQ para el alojamiento de juegos de bajo coste y en la nube en Amazon EC2 ya está disponible de forma general. Amazon GameLift FleetiQ ofrece a los desarrolladores la posibilidad de alojar servidores de juegos directamente en las instancias puntuales de Amazon EC2 al optimizar su viabilidad para el alojamiento de juegos. Los desarrolladores de juegos pueden usar Amazon GameLift FleetiQ con juegos nuevos o para complementar la capacidad de los juegos existentes. Esta solución admite el uso de contenedores u otros AWS servicios, como AWS Shield y Amazon Elastic Container Service (Amazon ECS).

Esta versión de disponibilidad general incluye las siguientes actualizaciones de la solución Amazon GameLift FleetiQ:

- La nueva operación de API `DescribeGameServerInstances` devuelve información, incluido el estado, de todas las instancias activas de un grupo de servidores de juegos de Amazon GameLift FleetiQ.
- La nueva estrategia de equilibrio, `ON_DEMAND_ONLY`, configura un grupo de servidores de juegos para que utilice únicamente instancias bajo demanda. Puede actualizar la estrategia de equilibrio de un grupo de servidores de juegos en cualquier momento, lo que permite cambiar entre el uso de instancias de spot e instancias bajo demanda según sea necesario.
- Se han eliminado los siguientes elementos de la vista previa por motivos de disponibilidad general:
  - Uso de claves de clasificación personalizadas para los recursos del servidor de juegos. Los servidores de juegos se pueden ordenar según la marca de tiempo de registro.
  - Etiquetado de los recursos del servidor de juegos.

16 de abril de 2020: Amazon GameLift actualiza el SDK del servidor para Unity y Unreal Engine

Versiones de SDK actualizadas: Amazon GameLift Server SDK 4.0.0, Amazon GameLift Local 1.0.5

La última versión del SDK de Amazon GameLift Server contiene los siguientes componentes actualizados:

- SDK de C#, versión 4.0.0 actualizada para Unity 2019.
- Complemento de Unreal, versión 3.3.1 actualizada para Unreal Engine, versiones 4.22, 4.23 y 4.24
- GameLift La versión 1.0.5 de Amazon Local se actualizó para probar las integraciones que utilizan la versión 4.0.0 del SDK del servidor C#.

Descarga la última versión del SDK de Amazon GameLift Server en Amazon GameLift Getting Starting.

2 de abril de 2020: Amazon GameLift FleetiQ está disponible para el alojamiento de juegos en EC2 (versión preliminar pública)

Versiones del SDK actualizadas: SDK 1.7.310 AWS

La función Amazon GameLift FleetiQ optimiza la viabilidad de las instancias puntuales de bajo coste para su uso con el alojamiento de juegos. Esta función ahora se amplía para los clientes que desean administrar sus recursos de alojamiento directamente en lugar de hacerlo a través del GameLift servicio gestionado de Amazon. Esta solución admite el uso de contenedores u otros AWS servicios, como AWS Shield y Amazon Elastic Container Service (Amazon ECS).

Más información:

GameTech entrada de blog en Amazon GameLift FleetiQ

19 de diciembre de 2019: administración de AWS recursos mejorada para GameLift los recursos de Amazon

Versiones del SDK actualizadas: AWS SDK 1.7.249

Ahora puede aprovechar las herramientas de administración de AWS recursos con Amazon GameLift Resources. En concreto, a todos los GameLift recursos clave de Amazon (compilaciones, scripts, flotas, colas de sesiones de juego, configuraciones de emparejamiento y conjuntos de reglas de emparejamiento) ahora se les asignan valores de nombre de recurso de Amazon (ARN). Un ARN de recurso proporciona un identificador coherente que es único en todas las AWS regiones. Se pueden usar para crear políticas de permisos para recursos específicos AWS Identity and Access Management (IAM). A los recursos se les asigna ahora un ARN, así como el identificador de recurso preexistente, que no es específico de la región.

Además, GameLift los recursos de Amazon ahora admiten el etiquetado. Puede utilizar etiquetas para organizar los recursos, crear políticas de permisos de IAM para gestionar el acceso a grupos de recursos, personalizar los desgloses de AWS costes, etc. Cuando gestione las etiquetas de GameLift los recursos de Amazon, utilice las acciones `TagResource()` de la GameLift API de Amazon y`ListTagsForResource()`. `UntagResource()`

Más información:

- TagResourceen la referencia de la GameLift API de Amazon
- Etiquetado de recursos de AWS en la Referencia general de AWS
- Nombres de recursos de Amazon en la Referencia general de AWS

14 de noviembre de 2019: Nuevas plantillas de AWS CloudFormation , actualizaciones en la región de China (Pekín)

Versiones del SDK actualizadas: AWS SDK 1.7.210

AWS CloudFormation plantillas para Amazon GameLift

GameLift Los recursos de Amazon ahora se pueden crear y gestionar a través de AWS CloudFormation. Las plantillas de AWS CloudFormation construcción y flota existentes se han actualizado para adaptarlas a los recursos actuales, y ahora hay nuevas plantillas disponibles para los guiones, las colas, las configuraciones de emparejamiento y los conjuntos de reglas de emparejamiento. AWS CloudFormation Las plantillas simplifican considerablemente la tarea de gestionar grupos de AWS recursos relacionados, especialmente cuando se despliegan juegos en varias regiones.

Más información:

- Referencia del tipo de GameLift recurso de Amazon en la Guía AWS CloudFormation del usuario
- Administración de recursos mediante AWS CloudFormationen la Guía para GameLift desarrolladores de Amazon

Glosario de AWS

Para ver la terminología más reciente de AWS, consulte el Glosario de AWS en la Referencia de Glosario de AWS.

Las traducciones son generadas a través de traducción automática. En caso de conflicto entre la traducción y la version original de inglés, prevalecerá la version en inglés.