aws

FlexMatch Leitfaden für Entwickler

Amazon GameLift

Version

Amazon GameLift: FlexMatch Leitfaden für Entwickler

Table of Contents

Was ist Amazon GameLift FlexMatch?

Amazon GameLift FlexMatch ist ein anpassbarer Matchmaking-Service für Multiplayer-Spiele. Mit FlexMatch können Sie ein benutzerdefiniertes Regelwerk erstellen, das definiert, wie ein Mehrspielerspiel für Ihr Spiel aussieht, und festlegt, wie kompatible Spieler für jedes Spiel bewertet und ausgewählt werden. Außerdem kannst du wichtige Aspekte des Matchmaking-Algorithmus an deine Spielanforderungen anpassen.

Verwenden Sie FlexMatch es als eigenständigen Matchmaking-Service oder integriert in eine GameLift Amazon-Game-Hosting-Lösung. Sie können es beispielsweise FlexMatch als eigenständige Funktion bei Spielen mit einer peer-to-peer Architektur oder bei Spielen, die andere Cloud-Computing-Lösungen verwenden, implementieren. Oder Sie können Ihr von Amazon GameLift verwaltetes EC2-Hosting oder Ihr lokales Hosting bei Amazon erweitern FlexMatch . GameLift Anywhere Dieser Leitfaden enthält detaillierte Informationen zum Aufbau eines FlexMatch Matchmaking-Systems für Ihr spezielles Szenario.

FlexMatch bietet Ihnen die Flexibilität, je nach Ihren Spielanforderungen Prioritäten für das Matchmaking festzulegen. Sie können z. B. Folgendes tun:

- Finden Sie ein Gleichgewicht zwischen Spielgeschwindigkeit und Qualität. Legen Sie Spielregeln fest, um schnell Spiele zu finden, die gut genug sind, oder lassen Sie die Spieler etwas länger warten, um das bestmögliche Spiel für ein optimales Spielerlebnis zu finden.
- Richten Sie Spiele auf der Grundlage von Spielern oder Teams aus, die gut zusammenpassen. Erstelle Spiele, bei denen alle Spieler ähnliche Eigenschaften wie Fähigkeiten oder Erfahrung haben. Oder bilden Sie Spiele, bei denen die kombinierten Eigenschaften jeder Mannschaft gemeinsame Kriterien erfüllen.
- Priorisieren Sie, wie die Latenz der Spieler beim Matchmaking berücksichtigt wird. Wollt ihr ein festes Latenzlimit für alle Spieler festlegen, oder sind höhere Latenzen akzeptabel, solange alle Spieler im Spiel eine ähnliche Latenz haben?

Bist du bereit, mit der Arbeit zu beginnen? FlexMatch

step-by-step Anleitungen dazu, wie Sie Ihr Spiel zum Laufen bringen können FlexMatch, finden Sie in den folgenden Themen:

- FlexMatchIntegration mit GameLift Amazon-Hosting

- GameLiftFlexMatchAmazon-Integration für eigenständiges Matchmaking

Die wichtigsten FlexMatch Funktionen

Die folgenden Funktionen sind in allen FlexMatch Szenarien verfügbar, unabhängig davon, ob Sie FlexMatch sie als eigenständigen Service oder mit GameLift Amazon-Spielehosting verwenden.

- Anpassbares Spieler-Matching. Entwirf und baue Matchmaker, die zu allen Spielmodi passen, die du deinen Spielern anbietest. Erstellen Sie eine Reihe von benutzerdefinierten Regeln, um wichtige Spielerattribute (wie Spielstärke oder Rolle) und geografische Latenzdaten zu bewerten, um großartige Spielermatches für Ihr Spiel zu erstellen.
- Latenzbasierter Abgleich. Stellen Sie Daten zur Spielerlatenz bereit und erstellen Sie Spielregeln, nach denen Spieler in einem Spiel ähnliche Reaktionszeiten haben müssen. Diese Funktion ist nützlich, wenn sich Ihre Spieler-Suchmaking-Pools über mehrere geografische Regionen erstrecken.
- Support für Spielgrößen von bis zu 200 Spielern. Erstelle Spiele mit bis zu 40 Spielern mithilfe von Spielregeln, die auf dein Spiel zugeschnitten sind. Erstellen Sie Spiele mit bis zu 200 Spielern mithilfe eines Matching-Prozesses, der einen optimierten benutzerdefinierten Matching-Prozess verwendet, um die Wartezeiten der Spieler überschaubar zu halten.
- Akzeptanz durch die Spieler. Fordere die Spieler auf, sich für ein geplantes Spiel anzumelden, bevor sie das Spiel beenden und eine Spielsitzung beginnen. Verwenden Sie diese Funktion, um Ihren benutzerdefinierten Annahme-Workflow zu starten und Spielerantworten zu melden, FlexMatch bevor Sie eine neue Spielsitzung für das Spiel platzieren. Wenn nicht alle Spieler ein Spiel annehmen, schlägt das vorgeschlagene Spiel fehl und Spieler, die es akzeptiert haben, kehren automatisch in den Matchmaking-Pool zurück.
- Unterstützung für Spielerparteien. Generieren Sie Spiele für Gruppen von Spielern, die zusammen in derselben Mannschaft spielen möchten. Verwenden Sie diese Option FlexMatch , um weitere Spieler zu finden, um das Spiel nach Bedarf auszufüllen.
- Erweiterbare Matching-Regeln. Lockern Sie die Spielanforderungen schrittweise, nachdem eine bestimmte Zeit vergangen ist, ohne dass ein erfolgreiches Spiel gefunden wurde. Durch die Erweiterung der Regeln können Sie entscheiden, wo und wann die ursprünglichen Spielregeln gelockert werden sollen, sodass die Spieler schneller mit spielbaren Spielen beginnen können.

- Auffüllen von Spielen. Füllen Sie die leeren Spielerplätze in einer bestehenden Spielsitzung mit gut passenden neuen Spielern. Passen Sie an, wann und wie neue Spieler angefordert werden sollen, und verwenden Sie dieselben benutzerdefinierten Spielregeln, um weitere Spieler zu finden.

FlexMatch mit GameLift Amazon-Hosting

FlexMatch bietet die folgenden zusätzlichen Funktionen für Spiele, die Sie bei Amazon hosten GameLift. Dazu gehören Spiele mit benutzerdefinierten Spieleservern oder Echtzeitservern.

- Platzierung der Spielsitzungen. Wenn ein Spiel erfolgreich abgeschlossen wurde, fordert Amazon FlexMatch automatisch eine neue Platzierung für eine Spielsitzung an GameLift. Die beim Matchmaking generierten Daten, einschließlich Spieler-IDs und Teamzuweisungen, werden dem Spieleserver zur Verfügung gestellt, sodass dieser diese Informationen verwenden kann, um die Spielsitzung für das Spiel zu starten. FlexMatch gibt dann Verbindungsinformationen zur Spielsitzung zurück, sodass Spieleclients dem Spiel beitreten können. Um die Latenz zu minimieren, die Spieler in einem Spiel erleben, GameLift kann die Platzierung von Spielsitzungen bei Amazon auch regionale Spielerlatenzdaten verwenden, sofern diese bereitgestellt werden.
- Automatisches Auffüllen von Spielen. Wenn diese Funktion aktiviert ist, FlexMatch wird automatisch eine Anfrage zum Auffüllen eines Matches gesendet, wenn eine neue Spielsitzung mit unbesetzten Spielerplätzen beginnt. Ihr Matchmaking-System startet den Platzierungsprozess für eine Spielsitzung mit einer Mindestanzahl von Spielern und füllt dann schnell die verbleibenden Plätze. Sie können das automatische Auffüllen nicht verwenden, um Spieler zu ersetzen, die aus einer Match-Spielsitzung aussteigen.

Wenn Sie Amazon GameLift FleetIQ mit Spielen verwenden, die mit Amazon Elastic Compute Cloud (Amazon EC2) -Ressourcen gehostet werden, implementieren Sie es FlexMatch als eigenständigen Service.

Preise für Amazon GameLift FlexMatch

Amazon GameLift berechnet für Instances nach Nutzungsdauer und für Bandbreite nach übertragener Datenmenge. Wenn Sie Ihre Spiele auf GameLift Amazon-Servern hosten, ist die FlexMatch Nutzung in den Gebühren für Amazon enthalten GameLift. Wenn Sie Ihre Spiele auf einer anderen Serverlösung hosten, wird die FlexMatch Nutzung separat berechnet. Eine vollständige Liste der Gebühren und Preise für Amazon finden Sie GameLift unter GameLift Amazon-Preise.

Informationen zur Berechnung der Kosten für das Hosten Ihrer Spiele oder das Matchmaking bei Amazon GameLift finden Sie unter Generieren von GameLift Amazon-Preisschätzungen, in dem beschrieben wird, wie Sie den AWS Pricing Calculatorverwenden.

So GameLift FlexMatch funktioniert Amazon

Dieses Thema bietet einen Überblick über den GameLift FlexMatch Amazon-Dienst, einschließlich der Kernkomponenten eines FlexMatch Systems und ihrer Interaktion.

Sie können es FlexMatch mit Spielen verwenden, die von Amazon GameLift Managed Hosting verwenden, oder mit Spielen, die eine andere Hosting-Lösung verwenden. Spiele, die auf GameLift Amazon-Servern, einschließlich Echtzeitservern, gehostet werden, verwenden den integrierten GameLift Amazon-Dienst, um automatisch verfügbare Spielserver zu finden und Spielsitzungen für die Spiele zu starten. Spiele, die FlexMatch als eigenständiger Service genutzt werden, einschließlich Amazon GameLift FleetIQ, müssen sich mit dem bestehenden Hosting-System abstimmen, um Hosting-Ressourcen zuzuweisen und Spielsitzungen für die Spiele zu starten.

Eine ausführliche Anleitung FlexMatch zur Einrichtung deiner Spiele findest du unterErste Schritte mit FlexMatch.

Matchmaking-Komponenten

Ein FlexMatch Matchmaking-System umfasst einige oder alle der folgenden Komponenten.

GameLiftAmazon-Komponenten

Dies sind GameLift Amazon-Ressourcen, die steuern, wie der FlexMatch Dienst das Matchmaking für Ihr Spiel durchführt. Sie werden mithilfe von GameLift Amazon-Tools, einschließlich der Konsole und der AWS CLI, oder alternativ programmgesteuert mithilfe des AWS SDK für Amazon erstellt und verwaltet. GameLift

- FlexMatchMatchmaking-Konfiguration (auch Matchmaker genannt) — Ein Matchmaker besteht aus einer Reihe von Konfigurationswerten, die den Matchmaking-Prozess für dein Spiel anpassen. Ein Spiel kann mehrere Matchmaker haben, die jeweils nach Bedarf für verschiedene Spielmodi oder Erlebnisse konfiguriert sind. Wenn dein Spiel eine Matchmaking-Anfrage an sendet, gibt es anFlexMatch, welcher Matchmaker verwendet werden soll.
- FlexMatchMatchmaking-Regelsatz — Ein Regelsatz enthält alle Informationen, die erforderlich sind, um Spieler für ein potenzielles Spiel zu bewerten und zu genehmigen oder abzulehnen.

Der Regelsatz definiert die Teamstruktur eines Spiels, legt die Spielerattribute fest, die für die Bewertung verwendet werden, und enthält Regeln, die die Kriterien für ein akzeptables Spiel beschreiben. Regeln können für einzelne Spieler, Teams oder das gesamte Spiel gelten. Eine Regel könnte zum Beispiel vorschreiben, dass alle Spieler im Spiel dieselbe Spielkarte wählen, oder sie könnte verlangen, dass alle Teams einen ähnlichen Fähigkeitsdurchschnitt haben.

- GameLiftAmazon-Spielsitzungswarteschlange (nur für FlexMatch von Amazon GameLift verwaltetes Hosting) — Eine Warteschlange für Spielsitzungen sucht nach verfügbaren Hosting-Ressourcen und startet eine neue Spielsitzung für das Spiel. Die Konfiguration der Warteschlange bestimmt, wo Amazon nach verfügbaren Hosting-Ressourcen GameLift sucht und wie der beste verfügbare Host für ein Match ausgewählt wird.

Benutzerdefinierte Komponenten

Die folgenden Komponenten umfassen Funktionen, die für ein vollständiges FlexMatch System erforderlich sind, das Sie basierend auf der Architektur Ihres Spiels implementieren müssen.

- Spielerschnittstelle für Matchmaking — Diese Oberfläche ermöglicht es Spielern, an einem Spiel teilzunehmen. Es initiiert mindestens eine Matchmaking-Anfrage über die Client-Matchmaking-Servicekomponente und stellt spielerspezifische Daten wie Fähigkeitsniveau und Latenzdaten bereit, die für den Matchmaking-Prozess benötigt werden.

  > **Note**
  >
  > Es hat sich bewährt, dass die Kommunikation mit dem FlexMatch Dienst über einen Backend-Service und nicht über einen Spieleclient erfolgen sollte.

- Client-Matchmaking-Service — Dieser Dienst beantwortet die Beitrittsanfragen des Spielers über die Spielerschnittstelle, generiert Matchmaking-Anfragen und sendet sie an den FlexMatch Dienst. Bei Anfragen, die gerade bearbeitet werden, überwacht es die Matchmaking-Ereignisse, verfolgt den Matchmaking-Status und ergreift bei Bedarf Maßnahmen. Je nachdem, wie du das Hosting von Spielsitzungen in deinem Spiel verwaltest, gibt dieser Dienst möglicherweise Verbindungsinformationen zu Spielsitzungen an die Spieler zurück. Diese Komponente verwendet das AWS SDK mit der GameLift Amazon-API, um mit dem FlexMatch Service zu kommunizieren.
- Spielplatzierungsservice (nur FlexMatch als eigenständiger Dienst) — Diese Komponente arbeitet mit Ihrem bestehenden Game-Hosting-System zusammen, um verfügbare Hosting-Ressourcen zu finden und neue Spielsitzungen für Spiele zu starten. Die Komponente muss die Matchmaking-Ergebnisse abrufen und die Informationen extrahieren, die für den Start einer neuen Spielsitzung

erforderlich sind, einschließlich Spieler-IDs, Attributen und Teamzuweisungen für alle Spieler im Spiel.

FlexMatchMatchmaking-Prozess

In diesem Thema werden ein grundlegendes Matchmaking-Szenario und Interaktionen zwischen den verschiedenen Spielkomponenten und dem FlexMatch Service beschrieben.

Matchmaking für Spieler anfragen

Ein Spieler, der deinen Spielclient verwendet, klickt auf die Schaltfläche „Spiel beitreten“. Diese Aktion veranlasst Ihren Client-Matchmaking-Service, eine Matchmaking-Anfrage an zu senden. FlexMatch Die Anfrage identifiziert den FlexMatch Matchmaker, der zur Erfüllung der Anfrage verwendet werden soll. Die Anfrage enthält auch Spielerinformationen, die Ihr benutzerdefinierter Matchmaker benötigt, wie z. B. Fähigkeitslevel, Spielpräferenzen oder geografische Latenzdaten. Sie können Matchmaking-Anfragen für einen oder mehrere Spieler stellen.

Anfragen zum Matchmaking-Pool hinzufügen

Wenn es die Matchmaking-Anfrage FlexMatch erhält, generiert es ein Matchmaking-Ticket und fügt es dem Ticketpool des Matchmakers hinzu. Das Ticket bleibt im Pool, bis es abgeglichen wird oder ein maximales Zeitlimit erreicht ist. Ihr Kunden-Matchmaking-Service wird regelmäßig über Matchmaking-Veranstaltungen informiert, einschließlich Änderungen des Ticketstatus.

Erstelle ein Match

Ihr FlexMatch Matchmaker führt kontinuierlich den folgenden Prozess für alle Tickets in seinem Pool aus:

1. Der Matchmaker sortiert den Pool nach dem Alter der Tickets und beginnt dann mit der Erstellung eines potenziellen Spiels, beginnend mit dem ältesten Ticket.
2. Der Matchmaker fügt dem potenziellen Spiel ein zweites Ticket hinzu und bewertet das Ergebnis anhand Ihrer benutzerdefinierten Matchmaking-Regeln. Wenn das potenzielle Spiel die Bewertung besteht, werden die Spieler des Tickets einer Mannschaft zugewiesen.
3. Der Matchmaker fügt nacheinander das nächste Ticket hinzu und wiederholt den Bewertungsprozess. Wenn alle Spielerplätze belegt sind, ist das Spiel bereit.

Bei der Spielersuche für große Matches (41 bis 200 Spieler) wird eine modifizierte Version des oben beschriebenen Prozesses verwendet, sodass Matches innerhalb eines angemessenen

Zeitrahmens erstellt werden können. Anstatt jedes Ticket einzeln zu bewerten, teilt der Matchmaker einen vorsortierten Ticketpool in potenzielle Spiele auf und gleicht dann jedes Spiel auf der Grundlage eines von Ihnen angegebenen Spielermerkmals aus. Ein Matchmaker kann zum Beispiel Tickets anhand ähnlicher Standorte mit niedriger Latenz vorsortieren und dann nach dem Spiel Balancing verwenden, um sicherzustellen, dass die Teams nach den Fähigkeiten der Spieler gleichmäßig verteilt sind.

Matchmaking-Ergebnisse melden

Wenn ein akzeptabler Treffer gefunden wird, werden alle übereinstimmenden Tickets aktualisiert und für jedes zugeordnete Ticket wird ein erfolgreiches Matchmaking-Event generiert.

- FlexMatchals eigenständiger Service: Ihr Spiel erhält Spielergebnisse bei einem erfolgreichen Matchmaking-Event. Die Ergebnisdaten beinhalten eine Liste aller gearteten Spieler und ihrer Teamzuweisungen. Wenn deine Spielanfragen Informationen zur Spielerlatenz enthalten, deuten die Ergebnisse auch auf einen optimalen geografischen Standort für das Spiel hin.
- FlexMatchmit einer GameLift Amazon-Hosting-Lösung: Die Spielergebnisse werden automatisch an eine GameLift Amazon-Warteschlange weitergeleitet, um die Spielsitzung zu platzieren. Der Matchmaker bestimmt, welche Warteschlange für die Platzierung der Spielsitzung verwendet wird.

Starte eine Spielsitzung für das Spiel

Nachdem ein geplantes Match erfolgreich gebildet wurde, wird eine neue Spielsitzung gestartet. Ihre Spielserver müssen in der Lage sein, die Matchmaking-Ergebnisdaten, einschließlich Spieler-IDs und Teamzuweisungen, bei der Einrichtung einer Spielsitzung für das Spiel zu verwenden.

- FlexMatchals eigenständiger Dienst: Ihr benutzerdefinierter Matchplatzierungsdienst ruft Spielergebnisdaten von erfolgreichen Matchmaking-Events ab und stellt eine Verbindung zu Ihrem bestehenden Platzierungssystem für Spielsitzungen her, um eine verfügbare Hosting-Ressource für das Spiel zu finden. Nachdem eine Hosting-Ressource gefunden wurde, koordiniert der Match Placement Service Ihr bestehendes Hosting-System, um eine neue Spielsitzung zu starten und Verbindungsinformationen zu erhalten.
- FlexMatchmit einer GameLift Amazon-Hosting-Lösung: Die Spielesitzungswarteschlange sucht den besten verfügbaren Spielserver für das Spiel. Je nachdem, wie die Warteschlange konfiguriert ist, versucht sie, die Spielsitzung mit den kostengünstigsten Ressourcen zu platzieren und die Spieler dort zu platzieren, wo die Latenz gering ist (sofern Daten zur Spielerlatenz bereitgestellt werden). Sobald die Spielsitzung erfolgreich platziert wurde, fordert der GameLift Amazon-Dienst den Spieleserver auf, eine neue Spielsitzung zu starten, wobei die Matchmaking-Ergebnisse und andere optionale Spieldaten weitergegeben werden.

Verbinde Spieler mit dem Spiel

Nachdem eine Spielsitzung gestartet wurde, stellen die Spieler eine Verbindung zu der Sitzung her, nehmen ihre Teamzuweisung in Anspruch und beginnen mit dem Spielen.

- FlexMatchals eigenständiger Dienst: Ihr Spiel verwendet das bestehende System zur Verwaltung von Spielsitzungen, um den Spielern Verbindungsinformationen zur Verfügung zu stellen.
- FlexMatchmit einer GameLift Amazon-Hosting-Lösung: Bei erfolgreicher Platzierung der Spielsitzung werden alle passenden Tickets mit Verbindungsinformationen zur Spielsitzung und einer Spielersitzungs-ID FlexMatch aktualisiert.

FlexMatch unterstützt AWS-Regionen

Wenn Sie FlexMatch mit einer Amazon GameLift -Hosting-Lösung verwenden, können Sie übereinstimmende Spielsitzungen an jedem Ort hosten, an dem Sie Spiele hosten. Sehen Sie sich die vollständige Liste der - AWS-Regionen und -Speicherorte für Amazon GameLift Hosting an.

Für alle FlexMatch Benutzer können Sie FlexMatch Ressourcen, einschließlich Matchmaking-Konfigurationen und Regelsätze, in den folgenden unterstützten hostenAWS-Regionen. Siehe Wählen Sie einen Speicherort für den Matchmaker.

AWS-Region-Name	Regionscode
USA Ost (Nord-Virginia)	us-east-1
USA West (Oregon)	us-west-2
Asien-Pazifik (Seoul)	ap-northest-2
Asien-Pazifik (Sydney)	ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1
Europa (Frankfurt)	eu-central-1
Europe (Ireland)	eu-west-1
China (Peking- und Ningxia-Regionen)	

Einrichten von FlexMatch

Amazon GameLift FlexMatch ist ein AWS Service, und Sie müssen über ein AWS Konto verfügen, um diesen Service nutzen zu können. Das Erstellen eines AWS-Kontos ist kostenlos. Weitere Informationen darüber, was Sie mit einem AWS Konto tun können, finden Sie unter Erste Schritte mit AWS.

Wenn Sie es FlexMatch zusammen mit anderen GameLift Amazon-Lösungen verwenden, lesen Sie die folgenden Themen:

- Zugriff für GameLift Amazon-Hosting und Echtzeitserver einrichten
- Zugriff für das Hosting auf Amazon EC2 mit Amazon GameLift FleetIQ einrichten

So richten Sie Ihr Konto für Amazon ein GameLift

1. Legen Sie ein Konto an. Öffnen Sie Amazon Web Services und wählen Sie In der Konsole anmelden. Folgen Sie den Anweisungen, um entweder ein neues Konto zu erstellen oder sich bei einem bestehenden anzumelden.
2. Richten Sie eine administrative Benutzergruppe ein. Öffnen Sie die AWS Identity and Access Management (IAM) Servicekonsole und folgen Sie den Schritten, um Benutzer oder Benutzergruppen zu erstellen oder zu aktualisieren. IAM verwaltet den Zugriff auf Ihre AWS Dienste und Ressourcen. Jeder, der über die GameLift Amazon-Konsole oder durch Aufrufen von GameLift Amazon-APIs auf Ihre FlexMatch Ressourcen zugreift, muss expliziten Zugriff erhalten. Detaillierte Anweisungen zur Verwendung der Konsole (oder der AWS CLI oder anderer Tools) zur Einrichtung von Benutzergruppen finden Sie unter IAM-Benutzer erstellen.
3. Fügen Sie Ihrem Benutzer oder Ihrer Benutzergruppe eine Berechtigungsrichtlinie hinzu.Der Zugriff auf AWS Dienste und Ressourcen wird verwaltet, indem einem Benutzer oder einer Benutzergruppe eine IAM-Richtlinie zugeordnet wird. Berechtigungsrichtlinien legen eine Reihe von AWS Diensten und Aktionen fest, auf die ein Benutzer Zugriff haben muss.

   Für Amazon GameLift müssen Sie eine benutzerdefinierte Berechtigungsrichtlinie erstellen und sie an jeden Benutzer oder jede Benutzergruppe anhängen. Eine Richtlinie ist ein JSON-Dokument. Verwenden Sie das folgende Beispiel, um Ihre Richtlinie zu erstellen.

Das folgende Beispiel veranschaulicht eine Inline-Berechtigungsrichtlinie mit Administratorberechtigungen für alle GameLift Amazon-Ressourcen und -Aktionen. Sie können den Zugriff einschränken, indem Sie nur FlexMatch bestimmte Elemente angeben.

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

Erste Schritte mit FlexMatch

Verwenden Sie die Ressourcen in diesem Abschnitt, um mit dem Aufbau eines Matchmaking-Systems zu beginnen. FlexMatch

Themen

GameLiftFlexMatchAmazon-Integration für eigenständiges Matchmaking

In diesem Thema wird der vollständige Integrationsprozess für die Implementierung FlexMatch als eigenständigen Matchmaking-Service beschrieben. Verwenden Sie diesen Prozesspeer-to-peer, wenn Ihr Multiplayer-Spiel mit benutzerdefinierter lokaler Hardware oder anderen Cloud-Rechensystemen gehostet wird. Dieser Prozess kann auch mit Amazon GameLift FleetIQ verwendet werden, einer Hosting-Optimierungslösung für Spiele, die auf Amazon EC2 gehostet werden. Wenn du dein Spiel mit von Amazon GameLift verwaltetem Hosting (einschließlich Echtzeitservern) hostest, findest du weitere Informationen unterFlexMatchIntegration mit GameLift Amazon-Hosting.

Bevor Sie mit der Integration beginnen, müssen Sie über ein AWS Konto verfügen und Zugriffsberechtigungen für den GameLift Amazon-Dienst einrichten. Details hierzu finden Sie unter Einrichten von FlexMatch. Alle wichtigen Aufgaben im Zusammenhang mit der Erstellung und Verwaltung von GameLift FlexMatch Amazon-Matchmakers und Regelsätzen können über die GameLift Amazon-Konsole erledigt werden.

1. Erstellen Sie einen FlexMatch Matchmaking-Regelsatz. Ihr benutzerdefinierter Regelsatz enthält vollständige Anweisungen zur Erstellung eines Spiels. Darin definieren Sie die Struktur und Größe jedes Teams. Sie geben auch eine Reihe von Anforderungen an, die ein Spiel erfüllen muss, um gültig zu sein. FlexMatch Dies bedeutet, Spieler in ein Spiel ein- oder auszuschließen. Diese Anforderungen können für einzelne Spieler gelten. Sie können den FlexMatch Algorithmus im Regelsatz auch anpassen, um beispielsweise große Matches mit bis zu 200 Spielern zu erstellen. Weitere Informationen finden Sie unter folgenden Themen:

   - Erstellen Sie einen FlexMatch Regelsatz
   - FlexMatch Beispiele für Regelsätze

2. Richten Sie Benachrichtigungen für Matchmaking-Events ein. Verwende Benachrichtigungen, um die FlexMatch Matchmaking-Aktivitäten zu verfolgen, einschließlich des Status ausstehender Spielanfragen. Dies ist der Mechanismus, der verwendet wird, um die Ergebnisse eines vorgeschlagenen Spiels zu liefern. Da die Matchmaking-Anfragen asynchron durchgeführt werden, benötigen Sie eine Möglichkeit, um den Status der Anforderungen nachzuverfolgen. Die Verwendung von Benachrichtigungen ist hierfür die bevorzugte Option. Weitere Informationen finden Sie unter folgenden Themen:

   - FlexMatchEventbenachrichtigungen einrichten
   - FlexMatchMatchmaking-Veranstaltungen

3. Richten Sie eine FlexMatch Matchmaking-Konfiguration ein. Diese Komponente wird auch als Matchmaker bezeichnet. Sie empfängt Matchmaking-Anfragen und verarbeitet sie. Sie konfigurieren einen Matchmaker, indem Sie einen Regelsatz, ein Benachrichtigungsziel und eine maximale Wartezeit angeben. Sie können auch optionale Funktionen aktivieren. Weitere Informationen finden Sie unter folgenden Themen:

   - Entwerfen eines FlexMatch Matchmakers
   - Erstellen Sie eine Matchmaking-Konfiguration

4. Bauen Sie einen Matchmaking-Service für Kunden auf. Erstelle oder erweitere einen Spiele-Client-Dienst mit Funktionen, an die du Matchmaking-Anfragen erstellen und senden kannst. FlexMatch Um Matchmaking-Anfragen erstellen zu können, muss diese Komponente über Mechanismen verfügen, mit denen die Spielerdaten abgerufen werden können, die gemäß dem Matchmaking-Regelsatz erforderlich sind, und optional Informationen zur regionalen Latenz. Es muss auch über eine Methode zum Erstellen und Zuweisen eindeutiger Ticket-IDs für jede Anfrage verfügen. Sie können sich auch dafür entscheiden, einen Workflow für die Spielerannahme einzurichten, bei dem sich die Spieler für ein geplantes Spiel anmelden müssen. Dieser Dienst muss auch Matchmaking-Ereignisse überwachen, um Spielergebnisse zu erhalten, und die Platzierung von Spielsitzungen für erfolgreiche Spiele veranlassen. Weitere Informationen finden Sie in diesem Thema:

   - FlexMatchZu einem Spielclient hinzufügen

5. Richten Sie einen Match-Vermittlungsdienst ein. Erstelle einen Mechanismus, der mit deinem bestehenden Game-Hosting-System zusammenarbeitet, um verfügbare Hosting-Ressourcen zu finden und neue Spielsitzungen für erfolgreiche Spiele zu starten. Diese Komponente muss in der Lage sein, Informationen über Spielergebnisse zu verwenden, um einen verfügbaren Spielserver abzurufen und eine neue Spielsitzung für das Spiel zu starten. Möglicherweise

möchten Sie auch einen Workflow implementieren, um Anfragen zum Auffüllen von Matches zu stellen. Dabei werden mithilfe von Matchmaking freie Plätze in bereits laufenden Spielsitzungen gefüllt.

FlexMatchIntegration mit GameLift Amazon-Hosting

FlexMatchist mit dem verwalteten GameLift Amazon-Hosting für benutzerdefinierte Spieleserver und Echtzeitserver verfügbar. Führen Sie die folgenden Aufgaben aus, um Ihrem Spiel FlexMatch-Matchmaking hinzuzufügen.

- Richten Sie einen Matchmaker ein. Ein Matchmaker empfängt Spieler-Matchmaking-Anforderungen und verarbeitet diese. Es gruppiert Spieler basierend auf definierten Regeln und erstellt für jeden erfolgreichen Match neue Spielsitzungen und Spielersitzungen. Gehen Sie wie folgt vor, um einen Matchmaker einzurichten:
  - Erstellen Sie einen Regelsatz. Ein Regelsatz zeigt dem Matchmaker, wie er gültige Matches erstellen soll. Sie gibt den Teamaufbau an und definiert, wie die Spieler im Hinblick auf die Aufnahme in ein Match ausgewertet werden. Weitere Informationen finden Sie unter folgenden Themen:
    - Erstellen Sie einen FlexMatch Regelsatz
    - FlexMatch Beispiele für Regelsätze
  - Erstellen Sie eine Spielsitzungs-Warteschlange. Eine Warteschlange sucht die beste Region für jedes Match und erstellt eine neue Spielsitzung in der betreffenden Region. Verwenden Sie für das Matchmaking eine vorhandene Warteschlange oder erstellen Sie eine neue Warteschlange. Weitere Informationen finden Sie in diesem Thema:
    - Erstellen Sie eine Warteschlange
  - Richten Sie Benachrichtigungen ein (optional). Da die Matchmaking-Anfragen asynchron durchgeführt werden, benötigen Sie eine Möglichkeit, um den Status der Anforderungen nachzuverfolgen. Benachrichtigungen sind die bevorzugte Option. Weitere Informationen finden Sie in diesem Thema:
    - FlexMatchEventbenachrichtigungen einrichten
  - Konfigurieren Sie einen Matchmaker. Nachdem Sie einen Regelsatz, eine Warteschlange und ein Benachrichtigungsziel festgelegt haben, erstellen Sie die Konfiguration für den Matchmaker. Weitere Informationen finden Sie unter folgenden Themen:
    - Entwerfen eines FlexMatch Matchmakers

    - Erstellen Sie eine Matchmaking-Konfiguration
- Integrieren Sie FlexMatch in Ihren Spiele-Client-Service. Fügen Sie Funktionen zu Ihrem Spiele-Client-Service hinzu, um Spilesitzungen mit Matchmaking zu starten. Matchmaking-Anforderungen geben an, welcher Matchmaker verwendet werden soll, und stellen die erforderlichen Spielerdaten für das Match bereit. Weitere Informationen finden Sie in diesem Thema:
    - FlexMatchZu einem Spielclient hinzufügen
- Integrieren Sie FlexMatch in Ihren Spielserver. Fügen Sie Funktionen zum Starten von Spielsitzungen zu Ihrem Spieleserver hinzu, die über Matchmaking erstellt werden. Anforderungen für diese Art von Spielsitzung umfassen matchspezifische Informationen, einschließlich Spieler und Teamzuweisungen. Der Spieleserver muss beim Erstellen einer Spielsitzung für das Match auf diese Informationen zugreifen und sie verwenden. Weitere Informationen finden Sie in diesem Thema:
    - FlexMatchZu einem von Amazon GameLift gehosteten Spieleserver hinzufügen
- Richten Sie FlexMatch-Backfill ein (optional). Fordern Sie zusätzliche Spieler-Matches an, um offene Spielerplätze in vorhandenen Spielen zu füllen. Sie können die automatische Auffüllung aktivieren, damit Amazon GameLift Backfill-Anfragen verwaltet. Alternativ können Sie Backfill manuell verwalten, indem Sie Funktionen zu Ihrem Spiele-Client-Service oder Spieleserver hinzufügen, um Match-Backfill-Anforderungen zu initiieren. Weitere Informationen finden Sie in diesem Thema:
    - Füllt bestehende Spiele auf mit FlexMatch

> **Note**
>
> FlexMatchBackfill ist derzeit nicht für Spiele verfügbar, die Echtzeitserver verwenden.

Aufbau eines Amazon-Matchmakers GameLift FlexMatch

Ein FlexMatch-Matchmaker-Prozess erstellt ein Spiele-Match. Er verwaltet die Pool der erhaltenen Matchmaking-Anforderungen, bildet Teams für ein Match, verarbeitet und wählt Spieler, um optimale Spielergruppen zu finden, und veranlasst den Prozess, eine Spielesitzung für das Match zu platzieren und zu starten. In diesem Thema werden die wichtigsten Aspekte eines Matchmakers beschreiben, und wie ein spezieller Matchmaker für Ihr Spiel konfiguriert wird.

Eine detaillierte Beschreibung der Verarbeitung der empfangenen Matchmaking-Anforderungen durch FlexMatch-Matchmaker finden Sie unter FlexMatchMatchmaking-Prozess.

Themen

Entwerfen eines FlexMatch Matchmakers

Dieses Thema enthält Anleitungen zum Entwerfen eines Matchmakers, der zu Ihrem Spiel passt.

Konfigurieren eines einfachen Matchmakers

Ein Matchmaker benötigt mindestens die folgenden Elemente:

- Der Regelsatz bestimmt die Größe und Umfang der Teams für ein Match und definiert einen Regelsatz, der für die Bewertung der Spieler für ein Match zu verwenden sind. Jeder Matchmaker wird so konfiguriert, dass er einen Regelsatz verwendet. Siehe Erstellen Sie einen FlexMatch Regelsatz und FlexMatch Beispiele für Regelsätze.
- Das Benachrichtigungsziel erhält alle Matchmaking-Ereignisbenachrichtigungen. Sie müssen ein Amazon Simple Notification Service (SNS)-Thema einrichten und dann die Themen-ID zum Matchmaker hinzufügen. Weitere Informationen zur Einrichtung von Benachrichtigungen finden Sie unter FlexMatchEventbenachrichtigungen einrichten.
- Das Anforderungs-Timeout bestimmt, wie lange Matchmaking-Anforderungen im Anforderungspool der Matchmaker bleiben und auf potenzielle Matches ausgewertet werden können. Wenn eine

Anforderung abgelaufen ist, ist kein Match zustande gekommen und sie wird aus dem Pool entfernt.

- Bei Verwendung von FlexMatch mit von Amazon GameLift verwaltetem Hosting findet die Spielsitzungswarteschlange die besten verfügbaren Ressourcen zum Hosten einer Spielsitzung für das Spiel und startet eine neue Spielsitzung. Jede Warteschlange ist mit einer Liste von Standorten und Ressourcentypen (einschließlich Spot- oder On-Demand-Instances) konfiguriert, die bestimmen, wo Spielsitzungen platziert werden können. Weitere Informationen zu Warteschlangen finden Sie unter Verwenden von Warteschlangen mit mehreren Standorten.

Wählen Sie einen Speicherort für den Matchmaker

Legen Sie fest, wo Matchmaking-Aktivitäten stattfinden sollen, und erstellen Sie Ihre Matchmaking-Konfiguration und Ihren Regelsatz an diesem Speicherort. Amazon GameLift verwaltet Ticketpools für die Match-Anforderungen Ihres Spiels, in denen sie sortiert und auf tragbare Matches hin bewertet werden. Nach dem Erstellen eines GameLift Matches sendet Amazon die Match-Details für die Platzierung von Spielsitzungen. Sie können die übereinstimmenden Spielsitzungen an jedem Ort ausführen, der von Ihrer Hosting-Lösung unterstützt wird.

Unter FlexMatch unterstützt AWS-Regionen finden Sie die Standorte, an denen Sie - FlexMatch Ressourcen erstellen können.

Berücksichtigen Sie bei der Auswahl eines AWS-Region für Ihren Matchmaker, wie sich der Standort auf die Leistung auswirken kann und wie er das Spielerlebnis für Spieler optimieren kann. Wir empfehlen Ihnen, die folgenden bewährten Methoden:

- Platzieren Sie einen Matchmaker an einem Ort, der sich in der Nähe Ihrer Spieler und Ihres Client-Services befindet, der FlexMatch Matchmaking-Anfragen sendet. Dieser Ansatz verringert die Latenzauswirkung auf Ihren Matchmaking-Anforderungsworkflow und macht ihn effizienter.
- Wenn Ihr Spiel ein globales Publikum erreicht, sollten Sie Matchmaker an mehreren Standorten erstellen und Match-Anfragen an den Matchmaker weiterleiten, der dem Spieler am nächsten ist. Dies erhöht nicht nur die Effizienz, sondern führt auch dazu, dass sich Ticketpools mit Spielern zusammenstellen, die geografisch nah beieinander sind, was die Fähigkeit des Spielers verbessert, Spieler auf der Grundlage der Latenzanforderungen zu vergleichen.
- Wenn Sie FlexMatch mit einem von Amazon GameLift verwalteten Hosting verwenden, platzieren Sie Ihren Matchmaker und die von ihm verwendete Spielsitzungswarteschlange am selben Ort. Dadurch können Sie die Kommunikation zwischen dem Matchmaker und der Warteschlange minimieren.

Hinzufügen optionaler Elemente

Zusätzlich zu diesen Mindestanforderungen können Sie Ihren Matchmaker mit den folgenden zusätzlichen Optionen konfigurieren. Wenn Sie FlexMatch mit einer Amazon GameLift -Hosting-Lösung verwenden, sind viele Funktionen integriert. Wenn Sie FlexMatch als eigenständigen Matchmaking-Service verwenden, sollten Sie diese Funktionen in Ihr System integrieren.

Spieler-Akzeptanz

Sie können einen Matchmaker so konfigurieren, dass alle Spieler, die für ein Spiel ausgewählt wurden, die Teilnahme akzeptieren müssen. Wenn Ihr System eine Annahme erfordert, müssen alle Spieler die Möglichkeit haben, ein vorgeschlagenes Match zu akzeptieren oder abzulehnen. Eine Match muss die Akzeptanz aller Spieler des vorgeschlagenen Match erhalten, bevor es fertiggestellt werden kann. Wenn ein Spieler ein Match ablehnt oder nicht akzeptiert, wird das vorgeschlagene Match verworfen und die Tickets werden wie folgt behandelt. Tickets, bei denen alle Spieler im Ticket das Spiel akzeptiert haben, werden zur weiteren Verarbeitung an den Matchmaking-Pool zurückgegeben. Tickets, bei denen mindestens ein Spieler das Match abgelehnt hat oder nicht reagiert hat, werden in einen Fehlerstatus versetzt und nicht mehr verarbeitet. Die Spielerakzeptanz erfordert ein Zeitlimit. Alle Spieler müssen ein vorgeschlagenes Match innerhalb des Zeitlimits akzeptieren, damit das Spiel fortgesetzt werden kann.

Backfill-Modus

Verwenden Sie FlexMatch Backfill, um Ihre Spielsitzungen während der gesamten Lebensdauer der Spielsitzung mit gut übereinstimmenden neuen Spielern zu füllen. Bei der Verarbeitung von Backfill-Anfragen FlexMatch verwendet denselben Matchmaker wie für den Abgleich mit den ursprünglichen Spielern. Sie können anpassen, wie Backfill-Tickets mit Tickets für neue Matches priorisiert werden, wobei Backfill-Tickets entweder an der Spitze oder am Ende der Zeile platziert werden. Das bedeutet, dass neue Spieler, die den Matchmaking-Pool betreten, mehr oder weniger wahrscheinlich in einem vorhandenen Spiel platziert werden als in einem neu gebildeten Spiel.

Manuelles Backfill ist verfügbar, unabhängig davon, ob Ihr Spiel FlexMatch mit verwaltetem Amazon GameLift -Hosting oder mit anderen Hosting-Lösungen verwendet. Manuelles Backfill bietet Ihnen die Flexibilität, zu entscheiden, wann eine Backfill-Anforderung ausgelöst werden soll. Beispielsweise möchten Sie neue Spieler möglicherweise nur in bestimmten Phasen Ihres Spiels oder nur dann hinzufügen, wenn bestimmte Bedingungen vorliegen.

Automatisches Backfill ist nur für Spiele verfügbar, die verwaltetes Amazon GameLift -Hosting verwenden. Wenn diese Funktion aktiviert ist und eine Spielsitzung mit offenen Spieler-Slots beginnt,

GameLift beginnt Amazon automatisch, Backfill-Anforderungen dafür zu generieren. Mit dieser Funktion können Sie Matchmaking so einrichten, dass neue Spiele mit einer Mindestanzahl von Spielern gestartet und dann schnell gefüllt werden, wenn neue Spieler in den Matchmaking-Pool aufgenommen werden. Sie können das automatische Backfill während der Spielsitzung jederzeit deaktivieren.

Spieleigenschaften

Für Spiele, die FlexMatch mit dem von Amazon GameLift verwalteten Hosting verwenden, können Sie zusätzliche Informationen angeben, die an einen Spieleserver übergeben werden, wenn eine neue Spielsitzung angefordert wird. Dies kann eine nützliche Methode sein, um Spielmoduskonfigurationen zu übergeben, die zum Starten einer Spielsitzung für die Art der zu erstellenden Spiele erforderlich sind. Alle Spielsitzungen für Matches, die von einem Matchmaker erstellt wurden, erhalten denselben Satz von Spieleigenschaften. Sie können die Informationen zu Spieleigenschaften ändern, indem Sie verschiedene Matchmaking-Konfigurationen erstellen.

Reservierte Spielerplätze

Sie können festlegen, dass bestimmte Spielerplätze in jedem Match reserviert und zu einem späteren Zeitpunkt gefüllt werden. Dies erfolgt, indem Sie die Eigenschaft für die „zusätzliche Spielerzahl“ einer Matchmaking-Konfiguration konfigurieren.

Benutzerdefinierte Ereignisdaten

Verwenden Sie diese Eigenschaft, um verschiedene benutzerdefinierte Informationen in alle mit dem Matchmaking verbundenen Ereignisse für den Matchmaker aufnehmen. Diese Funktion ist nützlich, um bestimmte Aktivitäten nachzuverfolgen, die einzigartig für Ihr Spiel sind, unter anderem die Leistung Ihrer Matchmaker.

Erstellen Sie einen FlexMatch Regelsatz

Jeder FlexMatch-Matchmaker benötigt einen Regelsatz. Der Regelsatz bestimmt die beiden Schlüsselelemente eines Matches: Team-Struktur und -Größe Ihres Spiels, und wie Spieler für ein bestmögliches Match zusammengruppiert werden.

Beispielsweise könnte ein Regelsatz ein Match wie folgt beschreiben: Erstelle ein Match mit zwei Teams mit je fünf Spielern, ein Team bildet die Verteidiger, das andere Team bildet die Angreifer. Ein Team kann Anfänger und erfahrene Spieler haben, aber die durchschnittliche Fähigkeit der beiden Teams muss innerhalb von 10 Punkten voneinander liegen. Wenn nach 30 Sekunden kein Match zustande kommt, lockern Sie die Qualifikationsanforderungen graduell.

Die Themen in diesem Abschnitt beschreiben, wie ein Matchmaking-Regelsatz entworfen und erstellt wird. Beim Erstellen eines Regelsatzes können Sie entweder die GameLift Amazon-Konsole oder die AWS CLI verwenden.

Topics

Entwerfen Sie einen FlexMatch Regelsatz

Dieses Thema behandelt die grundlegende Struktur eines Regelsatzes und wie man einen Regelsatz für kleine Spiele mit bis zu 40 Spielern erstellt. Ein Matchmaking-Regelsatz macht zwei Dinge: Sie legen die Teamstruktur und -größe eines Spiels fest und teilen Sie dem Matchmaker mit, wie er Spieler auswählt, um das bestmögliche Spiel zu bilden.

Ihr Matchmaking-Regelsatz kann jedoch mehr. Beispielsweise ist Folgendes möglich:

- Optimiere den Matchmaking-Algorithmus für dein Spiel.
- Lege Mindestanforderungen an die Spielerlatenz fest, um die Qualität des Gameplays zu gewährleisten.
- Lockere die Teamanforderungen und Spielregeln im Laufe der Zeit schrittweise, damit alle aktiven Spieler ein akzeptables Spiel finden können, wenn sie eines wollen.
- Definieren Sie die Behandlung von Gruppen-Matchmaking-Anfragen mithilfe der Gruppenaggregation.
- Verarbeite große Partien mit 40 oder mehr Spielern. Weitere Informationen zum Erstellen großer Streichhölzer finden Sie unterEntwerfen Sie einen FlexMatch Regelsatz für große Übereinstimmungen.

Beachten Sie beim Aufbau eines Matchmaking-Regelsatzes die folgenden optionalen und erforderlichen Aufgaben:

- Beschreiben Sie den Regelsatz (erforderlich)

- Passen Sie den Match-Algorithmus an
- Spielerattribute deklarieren
- Definiere Spielteams
- Lege Regeln für das Matching von Spielern fest
- Sorgen Sie dafür, dass sich die Anforderungen im Laufe der Zeit entspannen

Sie können Ihren Regelsatz mithilfe der GameLift Amazon-Konsole oder der `CreateMatchmakingRuleSet` Operation erstellen.

Beschreiben Sie den Regelsatz (erforderlich)

Machen Sie nähere Angaben zum Regelsatz.

- Name (optional) — Eine beschreibende Bezeichnung für Ihren eigenen Gebrauch. Dieser Wert ist nicht mit dem Namen des Regelsatzes verknüpft, den Sie bei der Erstellung des Regelsatzes mit Amazon angebenGameLift.
- ruleLanguageVersion— Die Version der Sprache für Eigenschaftsausdrücke, die zur Erstellung von FlexMatch Regeln verwendet wurde. Der Wert muss sein`1.0`.

Passen Sie den Match-Algorithmus an

FlexMatchoptimiert den Standardalgorithmus für die meisten Spiele, um die Spieler mit minimaler Wartezeit in akzeptable Matches zu bringen. Sie können den Algorithmus anpassen und das Matchmaking für Ihr Spiel anpassen.

Im Folgenden finden Sie den FlexMatch Standard-Matchmaking-Algorithmus:

1. FlexMatchplatziert alle offenen Matchmaking-Tickets und Backfill-Tickets in einen Ticketpool.
2. FlexMatchGruppiert Tickets im Pool nach dem Zufallsprinzip in einen oder mehrere Chargen. Wenn der Ticketpool größer wird, werden weitere Chargen FlexMatch gebildet, um die optimale Chargengröße beizubehalten.
3. FlexMatchsortiert die Tickets innerhalb jeder Charge nach Alter.
4. FlexMatcherstellt ein Match, das auf dem ältesten Ticket jeder Charge basiert.

Um den Match-Algorithmus anzupassen, fügen Sie Ihrem Regelsatzschema eine `algorithm` Komponente hinzu. Die vollständigen Referenzinformationen finden Sie unter. FlexMatchRegelsatzschema

Verwenden Sie die folgenden optionalen Anpassungen, um verschiedene Phasen Ihres Matchmaking-Prozesses zu beeinflussen.

- Sortierung vor dem Batch hinzufügen
- Bilden Sie Stapel auf der Grundlage von BatchDistance-Attributen
- Priorisieren Sie Backfill-Tickets
- Bevorzugen Sie ältere Tickets mit Erweiterungen

Sortierung vor dem Batch hinzufügen

Sie können den Ticketpool sortieren, bevor Sie Stapel bilden. Diese Art der Anpassung ist am effektivsten bei Spielen mit großen Ticketpools. Die Sortierung vor dem Batchvorgang kann dazu beitragen, den Matchmaking-Prozess zu beschleunigen und die Einheitlichkeit der Spieler bei den definierten Merkmalen zu erhöhen.

Definieren Sie Sortiermethoden vor dem Batch mithilfe der Algorithmuseigenschaft. `batchingPreference` Die Standardeinstellung lautet `random`.

Zu den Optionen für die Anpassung der Sortierung vor dem Batch gehören:

- Sortiere nach Spielerattributen. Geben Sie eine Liste der Spielerattribute an, um den Ticketpool vorab zu sortieren.

  Um nach Spielerattributen `batchingPreference` zu `sorted` sortieren, setzen Sie auf und definieren Sie Ihre Liste der Spielerattribute in`sortByAttributes`. Um ein Attribut zu verwenden, deklarieren Sie das Attribut zunächst in der `playerAttributes` Komponente des Regelsatzes.

  Im folgenden Beispiel wird der Ticketpool nach der bevorzugten Spielkarte der Spieler und dann nach den Fähigkeiten des Spielers FlexMatch sortiert. Es ist wahrscheinlicher, dass die resultierenden Batches ähnlich fähige Spieler enthalten, die dieselbe Karte verwenden möchten.

```
"algorithm": {
    "batchingPreference": "sorted",
    "sortByAttributes": ["map", "player_skill"],
    "strategy": "exhaustiveSearch"
```

```
},
```

- Sortiere nach Latenz. Erstellen Sie Matches mit der niedrigsten verfügbaren Latenz oder erstellen Sie schnell Matches mit akzeptabler Latenz. Diese Anpassung ist nützlich für Regelsätze, die große Matches mit mehr als 40 Spielern bilden.

  Stellen Sie die Algorithmuseigenschaft `strategy` auf ein`balanced`. Die ausgewogene Strategie begrenzt die verfügbaren Arten von Regelaussagen. Weitere Informationen finden Sie unter Entwerfen Sie einen FlexMatch Regelsatz für große Übereinstimmungen.

  FlexMatchsortiert Tickets auf der Grundlage der von Spielern gemeldeten Latenzdaten auf eine der folgenden Arten:

  - Standorte mit der niedrigsten Latenz. Der Ticketpool ist vorsortiert nach den Orten, an denen die Spieler ihre niedrigsten Latenzwerte angeben. FlexMatchfasst dann Tickets mit geringer Latenz an denselben Orten zusammen und sorgt so für ein besseres Spielerlebnis. Es reduziert auch die Anzahl der Tickets in jedem Stapel, sodass das Matchmaking länger dauern kann. Um diese Anpassung zu verwenden, setzen Sie `batchingPreferencefastestRegion`, wie im folgenden Beispiel gezeigt, auf.

    ```
    "algorithm": {
        "batchingPreference": "fastestRegion",
        "strategy": "balanced"
    },
    ```

  - Die akzeptable Latenz passt schnell. Der Ticketpool ist vorsortiert nach Orten, an denen Spieler einen akzeptablen Latenzwert angeben. Dadurch entstehen weniger Stapel mit mehr Tickets. Mit mehr Tickets in jedem Stapel ist es schneller, akzeptable Treffer zu finden. Um diese Anpassung zu verwenden, setzen Sie die Eigenschaft `batchingPreference` auf `largestPopulation`, wie im folgenden Beispiel gezeigt.

    ```
    "algorithm": {
        "batchingPreference": "largestPopulation",
        "strategy": "balanced"
    },
    ```

  > **Note**
  >
  > Der Standardwert für die ausgewogene Strategie ist`largestPopulation`.

Priorisieren Sie Backfill-Tickets

Wenn dein Spiel automatisches oder manuelles Auffüllen implementiert, kannst du je nach Art der Anfrage anpassen, wie Matchmaking-Tickets FlexMatch verarbeitet werden. Der Anforderungstyp kann eine neue Match- oder Backfill-Anfrage sein. Standardmäßig werden beide Arten von Anfragen gleich FlexMatch behandelt.

Die Priorisierung von Backfill-Prioritäten wirkt sich darauf aus, wie Tickets FlexMatch behandelt werden, nachdem sie gebündelt wurden. Die Priorisierung von Backfill-Prioritäten erfordert, dass Regelsätze die umfassende Suchstrategie anwenden.

FlexMatchordnet nicht mehrere Backfill-Tickets zusammen.

Um die Priorisierung für Backfill-Tickets zu ändern, legen Sie die Eigenschaft fest. `backfillPriority`

- Ordnen Sie zuerst die Backfill-Tickets zu. Diese Option versucht, Backfill-Tickets abzugleichen, bevor neue Treffer erstellt werden. Das bedeutet, dass neue Spieler eine höhere Chance haben, einem bestehenden Spiel beizutreten.

  Es ist am besten, dies zu verwenden, wenn dein Spiel Auto-Backfill verwendet. Das automatische Auffüllen wird häufig in Spielen mit kurzen Spielsitzungen und hohem Spieler-Turnaround verwendet. Das automatische Auffüllen hilft diesen Spielen dabei, möglichst wenige Matches zu bilden und sie zu starten, während FlexMatch nach weiteren Spielern gesucht wird, um freie Plätze zu besetzen.

  Legen Sie den Wert für `backfillPriority` auf `high` fest.

  ```
  "algorithm": {
      "backfillPriority": "high",
      "strategy": "exhaustiveSearch"
  },
  ```

- Spielen Sie die Backfill-Tickets zuletzt ab. Diese Option ignoriert Backfill-Tickets, bis alle anderen Tickets ausgewertet werden. Das bedeutet, dass FlexMatch neue Spieler in bestehende Spiele einsteigen, wenn sie nicht in neue Spiele aufgenommen werden können.

  Diese Option ist nützlich, wenn du das Auffüllen als letzte Chance nutzen möchtest, um Spieler für ein Spiel zu gewinnen, z. B. wenn nicht genügend Spieler für ein neues Spiel vorhanden sind.

  Setzen Sie `backfillPriority` auf `low`.

```
"algorithm": {
    "backfillPriority": "low",
    "strategy": "exhaustiveSearch"
},
```

Bevorzugen Sie ältere Tickets mit Erweiterungen

Die Erweiterungsregeln lockern die Spielkriterien, wenn Spiele schwierig zu beenden sind. Amazon GameLift wendet die Erweiterungsregeln an, wenn Tickets für ein teilweise abgeschlossenes Spiel ein bestimmtes Alter erreichen. Die Erstellungszeitstempel der Tickets bestimmen, wann Amazon die Regeln GameLift anwendet. Standardmäßig wird der Zeitstempel des zuletzt zugeordneten Tickets FlexMatch protokolliert.

Um zu ändern, wann Erweiterungsregeln FlexMatch angewendet werden, legen Sie die Eigenschaft `expansionAgeSelection` wie folgt fest:

- Erweitern Sie anhand der neuesten Tickets. Diese Option wendet Erweiterungsregeln an, die auf dem neuesten Ticket basieren, das dem potenziellen Spiel hinzugefügt wurde. Jedes Mal, FlexMatch wenn ein neues Ticket gefunden wird, wird die Uhr zurückgesetzt. Mit dieser Option sind die Ergebnisse in der Regel qualitativ hochwertiger, aber es dauert länger, bis sie abgeglichen werden. Match-Anfragen können zu lange dauern, bis sie abgeschlossen sind. Stellen Sie `expansionAgeSelection` auf`newest`. `newest`ist Standard.
- Erweitern Sie anhand der ältesten Tickets. Diese Option wendet Erweiterungsregeln an, die auf dem ältesten Ticket im potenziellen Spiel basieren. Mit dieser Option werden Erweiterungen schneller FlexMatch angewendet, wodurch die Wartezeiten für die Spieler, die am frühesten gefunden wurden, verbessert, aber die Spielqualität für alle Spieler sinkt. Setzen Sie `expansionAgeSelection` auf `oldest`.

```
"algorithm": {
    "expansionAgeSelection": "oldest",
    "strategy": "exhaustiveSearch"
},
```

Spielerattribute deklarieren

In diesem Abschnitt listet ihr die Attribute einzelner Spieler auf, die in Matchmaking-Anfragen berücksichtigt werden sollen. Es gibt zwei Gründe, warum du Spielerattribute in einem Regelsatz deklarieren könntest:

- Wenn der Regelsatz Regeln enthält, die von Spielerattributen abhängen.
- Wenn Sie ein Spielerattribut über die Spielanfrage an die Spielsitzung weitergeben möchten. Du könntest zum Beispiel die Charakterauswahl eines Spielers an die Spielsitzung weitergeben, bevor jeder Spieler eine Verbindung herstellt.

Beim Deklarieren eines Spielerattributs geben Sie die folgenden Informationen an:

- name (erforderlich) — Dieser Wert muss für den Regelsatz eindeutig sein.
- type (erforderlich) — Der Datentyp des Attributwerts. Gültige Datentypen sind Zahl, Zeichenfolge oder Zeichenfolgenzuweisung.
- default (optional) — Geben Sie einen Standardwert ein, der verwendet werden soll, wenn eine Matchmaking-Anfrage keinen Attributwert bereitstellt. Wenn kein Standard deklariert ist und eine Anfrage keinen Wert enthält, FlexMatch kann die Anfrage nicht erfüllt werden.

Definiere Spielteams

Beschreiben Sie die Struktur und Größe der Teams für ein Match. Jedes Match muss über mindestens ein Team verfügen, und Sie können beliebig viele Teams definieren. Ihre Teams können die gleiche Anzahl von Spielern haben oder asymmetrisch sein. Sie definieren z. B. möglicherweise ein Singleplayer-Monster-Team und ein Jäger-Team mit 10 Spielern.

FlexMatch verarbeitet Match-Anforderungen entweder als kleines Match oder als großes Match, je nachdem, wie Teamgrößen im Regelsatz definiert sind. Mögliche Matches mit bis zu 40 Spielern sind kleine Matches, Matches mit mehr als 40 Spielern sind große Matches. Um die potenzielle Match-Größe eines Regelsatzes zu bestimmen, addieren Sie die maxPlayer-Einstellungen für alle im Regelsatz definierten Teams.

- Name (erforderlich) — Weisen Sie jedem Team einen eindeutigen Namen zu. Du verwendest diesen Namen in Regeln und Erweiterungen sowie in FlexMatch Referenzen für die Matchmaking-Daten in einer Spielsitzung.

- MaxPlayers (erforderlich) — Geben Sie die maximale Anzahl von Spielern an, die dem Team zugewiesen werden sollen.
- MinPlayers (erforderlich) — Geben Sie die Mindestanzahl an Spielern an, die dem Team zugewiesen werden sollen.
- Menge (optional) — Geben Sie die Anzahl der Teams an, die mit dieser Definition gebildet werden sollen. Wenn ein Spiel FlexMatch erstellt wird, erhalten diese Teams den angegebenen Namen mit einer angehängten Nummer. Zum Beispiel `Red-Team1Red-Team2`, und`Red-Team3`.

FlexMatchversucht, Teams bis zur maximalen Spielergröße zu besetzen, bildet aber Teams mit weniger Spielern. Wenn Sie möchten, dass alle Teams im Match die gleiche Größe haben, können Sie eine Regel dafür erstellen. Ein Beispiel für eine `EqualTeamSizes` Regel finden Sie im FlexMatch Beispiele für Regelsätze Thema.

Lege Regeln für das Matching von Spielern fest

Erstellen Sie eine Reihe von Regelerklärungen, in denen die Spieler bewertet werden, ob sie in ein Spiel aufgenommen werden. Regeln können Anforderungen festlegen, die für einzelne Spieler, Teams oder ein gesamtes Match gelten. Wenn Amazon eine Spielanfrage GameLift bearbeitet, beginnt es mit dem ältesten Spieler im Pool der verfügbaren Spieler und erstellt ein Match um diesen Spieler herum. Ausführliche Hilfe zum Erstellen von FlexMatch Regeln finden Sie unterFlexMatchRegeltypen.

- name (erforderlich) — Ein aussagekräftiger Name, der die Regel innerhalb eines Regelsatzes eindeutig identifiziert. Regelnamen werden auch in Ereignisprotokollen und Metriken verwendet, in denen die Aktivitäten im Zusammenhang mit dieser Regel verfolgt werden.
- Beschreibung (optional) — Verwenden Sie dieses Element, um eine Freitextbeschreibung anzuhängen.
- type (erforderlich) — Das Typ-Element identifiziert die Operation, die bei der Verarbeitung der Regel verwendet werden soll. Jeder Regeltyp erfordert eine Reihe von zusätzlichen Eigenschaften. Eine Liste gültiger Regeltypen und Eigenschaften finden Sie unter FlexMatchRegeln, Sprache.
- Regeltypeigenschaft (möglicherweise erforderlich) — Abhängig vom definierten Regeltyp müssen Sie möglicherweise bestimmte Regeleigenschaften festlegen. Weitere Informationen zu Eigenschaften und zur Verwendung der FlexMatch-Eigenschaftsausdruckssprache finden Sie unter FlexMatchRegeln, Sprache.

Sorgen Sie dafür, dass sich die Anforderungen im Laufe der Zeit entspannen

Erweiterungen ermöglichen es dir, die Regelkriterien im Laufe der Zeit zu lockern, wenn FlexMatch du keinen Treffer findest. Diese Funktion stellt sicher, dass FlexMatch eine Bestleistung verfügbar ist, wenn sie nicht perfekt zusammenpasst. Indem Sie Ihre Regeln mit einer Erweiterung lockern, erweitern Sie nach und nach den Spielerpool, der zu einem akzeptablen Spiel passt.

Erweiterungen beginnen, wenn das Alter des neuesten Tickets im unvollständigen Spiel einer Wartezeit für die Erweiterung entspricht. Wenn FlexMatch dem Spiel ein neues Ticket hinzugefügt wird, wird die Wartezeit für die Erweiterung möglicherweise zurückgesetzt. Sie können im `algorithm` Abschnitt des Regelsatzes anpassen, wie Erweiterungen beginnen.

Hier ist ein Beispiel für eine Erweiterung, die die für das Spiel erforderliche Mindestqualifikationsstufe schrittweise erhöht. Der Regelsatz verwendet eine Abstandsregel, die so benannt SkillDeltaist, dass alle Spieler in einem Spiel innerhalb von 5 Fähigkeitsstufen voneinander entfernt sein müssen. Wenn fünfzehn Sekunden lang keine neuen Spiele gespielt werden, sucht diese Erweiterung nach einem Unterschied von 10 und zehn Sekunden später nach einem Unterschied von 20.

```
"expansions": [{
        "target": "rules[SkillDelta].maxDistance",
        "steps": [{
            "waitTimeSeconds": 15,
            "value": 10
        }, {
            "waitTimeSeconds": 25,
            "value": 20
        }]
    }]
```

Wenn Matchmaker die automatische Auffüllung aktiviert haben, sollten Sie Ihre Anforderungen an die Spieleranzahl nicht zu schnell lockern. Es dauert einige Sekunden, bis die neue Spielsitzung startet und mit dem automatischen Backfill beginnt. Ein besserer Ansatz besteht darin, mit der Erweiterung zu beginnen, nachdem das automatische Auffüllen für Ihre Spiele in der Regel aktiviert wird. Der Zeitpunkt der Erweiterung hängt von der Zusammensetzung deines Teams ab. Teste also, um die beste Erweiterungsstrategie für dein Spiel zu finden.

Entwerfen Sie einen FlexMatch Regelsatz für große Übereinstimmungen

Wenn dein Regelsatz Matches vorsieht, die 41 bis 200 Spieler zulassen, musst du einige Anpassungen an deiner Regelsatzkonfiguration vornehmen. Diese Anpassungen optimieren den

Spielalgorithmus, sodass er tragfähige große Matches erstellen und gleichzeitig die Wartezeiten der Spieler kurz halten kann. Daher werden zeitaufwändige benutzerdefinierte Regeln durch umfangreiche Match-Regelsätze durch Standardlösungen ersetzt, die für gängige Matchmaking-Prioritäten optimiert sind.

So ermittelst du, ob du deinen Regelsatz für große Matches optimieren musst:

1. Ermitteln Sie für jedes Team, das in Ihrem Regelsatz definiert ist, den Wert von MaxPlayer,
2. Addieren Sie alle MaxPlayer-Werte. Wenn die Gesamtzahl 40 übersteigt, haben Sie einen großen Spielregelsatz.

Um deinen Regelsatz für große Matches zu optimieren, nimm die folgenden Anpassungen vor. Das Schema für einen großen Spielregelsatz finden Sie unter Regelsatzschema für große Treffer und Beispiele für Regelsätze unterBeispiel 7: Erstellen einer großen Übereinstimmung.

Passen Sie den Match-Algorithmus für große Matches an

Fügen Sie dem Regelsatz eine Algorithmuskomponente hinzu, falls noch keine vorhanden ist. Stellen Sie die folgenden Eigenschaften ein.

- `strategy`(erforderlich) — Setzen Sie die `strategy` Eigenschaft auf „ausgewogen“. Mit dieser Einstellung werden FlexMatch nach dem Spiel zusätzliche Prüfungen durchgeführt, um anhand eines bestimmten Spielerattributs, das in der `balancedAttribute` Eigenschaft definiert ist, die optimale Teambalance zu finden. Die ausgewogene Strategie ersetzt die Notwendigkeit benutzerdefinierter Regeln zur Bildung gleichberechtigter Teams.
- `balancedAttribute`(erforderlich) — Identifizieren Sie ein Spielerattribut, das beim Balancieren der Teams in einem Spiel verwendet werden soll. Dieses Attribut muss einen numerischen Datentyp haben (double oder integer). Wenn Sie sich zum Beispiel für ein ausgewogenes Verhältnis zwischen den Fähigkeiten der Spieler entscheiden, FlexMatch versuchen Sie, die Spieler so zuzuweisen, dass alle Teams über möglichst gleiche Fähigkeitsstufen verfügen. Das Balancing-Attribut muss in den Spielerattributen des Regelsatzes deklariert werden.
- `batchingPreference`(optional) — Wähle aus, wie viel Wert du darauf legen möchtest, Matches mit möglichst niedriger Latenz für deine Spieler zu veranstalten. Diese Einstellung wirkt sich darauf aus, wie Spieltickets vor der Erstellung von Spielen sortiert werden. Zu den Optionen gehören:
  - Größte Bevölkerung. FlexMatcherlaubt Matches, bei denen alle Tickets im Pool verwendet werden, die akzeptable Latenzwerte an mindestens einem Ort gemeinsam haben. Infolgedessen ist der potenzielle Ticketpool in der Regel groß, was es einfacher macht, Spiele schneller zu

besetzen. Spieler werden möglicherweise in Spiele mit akzeptabler, aber nicht immer optimaler Latenz versetzt. Wenn die `batchingPreference` Eigenschaft nicht gesetzt ist, ist dies das Standardverhalten, wenn sie auf „ausgewogen“ gesetzt `strategy` ist.

- Schnellster Standort. FlexMatchSortiert alle Tickets im Pool danach, wo sie die niedrigsten Latenzwerte melden. Daher werden Matches in der Regel mit Spielern ausgetragen, die von einer geringen Latenz an denselben Orten berichten. Gleichzeitig ist der potenzielle Ticketpool für jedes Spiel kleiner, was die Zeit verlängern kann, die benötigt wird, um ein Spiel zu füllen. Da der Latenz eine höhere Priorität eingeräumt wird, können sich die Spieler in Spielen außerdem hinsichtlich des Balancing-Attributs stärker unterscheiden.

Im folgenden Beispiel wird der Spielalgorithmus so konfiguriert, dass er sich wie folgt verhält: (1) Sortiert den Ticketpool vor, um Tickets nach Standort zu gruppieren, an dem sie akzeptable Latenzwerte haben; (2) Bilden Sie Stapel sortierter Tickets für den Abgleich; (3) Erstellen Sie Matches mit Tickets in einem Stapel und gleichen Sie die Teams aus, um die durchschnittliche Spielerfähigkeit auszugleichen.

```
"algorithm": {
    "strategy": "balanced",
    "balancedAttribute": "player_skill",
    "batchingPreference": "largestPopulation"
},
```

Spielerattribute deklarieren

Stellen Sie sicher, dass Sie das Spielerattribut deklarieren, das im Regelsatzalgorithmus als Ausgleichsattribut verwendet wird. Dieses Attribut sollte für jeden Spieler in einer Matchmaking-Anfrage enthalten sein. Sie können einen Standardwert für das Spielerattribut angeben, aber der Attributausgleich funktioniert am besten, wenn spielerspezifische Werte angegeben werden.

Teams definieren

Der Prozess zum Definieren der Teamgröße und -struktur gestaltet sich genauso wie bei kleinen Matches, außer dass FlexMatch Teams auf unterschiedliche Weise füllt. Dies wirkt sich darauf aus, wie Matches wahrscheinlich aussehen, wenn sie nur teilweise gefüllt sind. Möglicherweise möchten Sie als Reaktion darauf Ihre Mindestteamgröße anpassen.

FlexMatch weist einen Spieler nach folgenden Regeln einem Team zu. Erstens: Suchen Sie nach Teams, die die minimal erforderliche Spieler-Anzahl noch nicht erreicht haben. Zweitens: Suchen Sie unter diesen Teams nach dem Team mit den meisten noch nicht gefüllten Spielerplätzen.

Bei Matches, die mehrere Teams gleicher Größe definieren, werden Spieler jedem Team sequenziell hinzugefügt, bis sie voll sind. Infolgedessen haben die Teams in einem Spiel immer eine fast gleiche Anzahl von Spielern, auch wenn das Spiel nicht voll ist. Derzeit gibt es keine Möglichkeit, bei großen Matches Teams gleicher Größe zu erzwingen. Bei Matches mit asymmetrisch großen Teams ist der Prozess etwas komplexer. In diesem Szenario werden die Spieler zunächst den größten Teams zugewiesen, die die meisten offenen Plätze haben. Da die Anzahl der offenen Plätze gleichmäßiger auf alle Teams verteilt wird, werden die Spieler in die kleineren Teams aufgeteilt.

Nehmen wir zum Beispiel an, Sie haben einen Regelsatz mit drei Teams. Die roten und blauen Teams sind beide auf `maxPlayers` =10, `minPlayers` =5 gesetzt. Das grüne Team ist auf `maxPlayers` =3, `minPlayers` =2 gesetzt. Hier ist die Füllsequenz:

1. Kein Team hat es erreicht`minPlayers`. Das rote und das blaue Team besitzen 10 verfügbare Spielerplätze und das grüne Team hat 3. Die ersten 10 Spieler (jeweils 5) werden dem roten und dem blauen Team zugewiesen. Beide Teams haben es jetzt erreicht`minPlayers`.
2. Das grüne Team ist noch nicht erreicht`minPlayers`. Die nächsten 2 Spieler werden dem grünen Team zugewiesen. Das grüne Team ist jetzt angekommen`minPlayers`.
3. Da alle Teams am Start sind`minPlayers`, werden nun zusätzliche Spieler auf der Grundlage der Anzahl der offenen Plätze zugewiesen. Das rote und das blaue Team haben jeweils 5 offene Plätze, während das grüne Team 1 hat. Die nächsten 8 Spieler werden (jeweils 4) den Teams Rot und Blau zugewiesen. Alle Teams haben jetzt 1 freien Platz.
4. Die verbleibenden 3 Spielerplätze (jeweils 1) werden Teams in keiner bestimmten Reihenfolge zugewiesen.

Lege Regeln für große Spiele fest

Das Matchmaking für große Spiele hängt hauptsächlich von der Balancing-Strategie und den Batching-Optimierungen der Latenz ab. Die meisten benutzerdefinierten Regeln sind nicht verfügbar. Sie können jedoch die folgenden Regeltypen integrieren:

- Regel, die eine feste Grenze für die Latenz von Spielern festlegt. Verwenden Sie den `latency` Regeltyp mit der Eigenschaft`maxLatency`. Siehe Latenz-Regel Referenz. Im folgenden Beispiel wird die maximale Spieler-Latenzwert auf 200 Millisekunden eingestellt:

```
"rules": [{
        "name": "player-latency",
        "type": "latency",
        "maxLatency": 200
```

```
    }],
```

- Regel, nach der Spieler auf der Grundlage ihrer Nähe in einem bestimmten Spielerattribut aufgeteilt werden. Dies unterscheidet sich von der Definition eines Ausgleichsattributs im Rahmen des Algorithmus für große Spiele, bei dem der Schwerpunkt auf der Bildung gleichmäßiger Teams liegt. Bei dieser Regel werden Matchmaking-Tickets auf der Grundlage der Ähnlichkeit der angegebenen Attributwerte, wie z. B. der Fähigkeit eines Anfängers oder eines Experten, gebündelt, was in der Regel dazu führt, dass Spieler in Bezug auf das angegebene Attribut eng aufeinander abgestimmt sind. Verwenden Sie den `batchDistance` Regeltyp, identifizieren Sie ein numerisch basiertes Attribut und geben Sie den größtmöglichen zulässigen Bereich an. Siehe Regel für den Batch-Abstand Referenz. Hier ist ein Beispiel, das verlangt, dass sich die Spieler eines Spiels innerhalb einer Fähigkeitsstufe voneinander befinden:

```
"rules": [{
        "name": "batch-skill",
        "type": "batchDistance",
        "batchAttribute": "skill",
        "maxDistance": 1
```

Relaxen Sie große Spielanforderungen

Sie können genauso wie bei kleinen Matches Match-Anforderungen mithilfe von Erweiterungen mit der Zeit lockern, wenn andernfalls keine gültigen Matches möglich sind. Bei großen Matches hast du die Möglichkeit, entweder die Latenzregeln zu lockern oder die Anzahl der Teamspieler zu lockern.

Wenn du die automatische Auffüllung von Matches für große Spiele verwendest, vermeide es, die Anzahl deiner Teamspieler zu schnell zu verringern. FlexMatchbeginnt erst nach dem Start einer Spielsitzung mit der Generierung von Backfill-Anfragen. Dies kann einige Sekunden nach der Erstellung eines Matches nicht der Fall sein. Während dieser Zeit kann FlexMatch mehrere teilweise gefüllte neue Spielsitzungen erstellen, insbesondere wenn die Regeln bezüglich der Spieleranzahl gelockert werden. Dies kann letztendlich in mehr Spielsitzungen resultieren, als Sie benötigen, mit zu wenigen Spielern pro Spiel. Es ist eine bewährte Methode, dem ersten Schritt in der Erweiterung der Spieleranzahl eine längere Wartezeit zuzuweisen, während der Ihre Spielsitzung gestartet werden kann. Da Backfill-Anforderungen bei größeren Matches eine höhere Priorität zukommt, werden eingehende Spieler in bestehende Spielen platziert, bevor ein neues Spiel gestartet wird. Möglicherweise müssen Sie experimentieren, um die ideale Wartezeit für Ihr Spiel zu finden.

Es folgt ein Beispiel, in dem die Spieleranzahl des gelben Teams mit einer längeren anfänglichen Wartezeit graduell verringert wird. Denken Sie daran, dass Wartezeiten in Regelsatz-Erweiterungen absolut sind und nicht summiert werden. Die erste Erweiterung tritt bei fünf Sekunden auf und die zweite Erweiterung fünf Sekunden später, also bei zehn Sekunden.

```
"expansions": [{
        "target": "teams[Yellow].minPlayers",
        "steps": [{
            "waitTimeSeconds": 5,
            "value": 8
        }, {
            "waitTimeSeconds": 10,
            "value": 5
        }]
    }]
```

Erstellen von Matchmaking-Regelsätzen

Bevor Sie einen Matchmaking-Regelsatz für Ihren Amazon GameLift FlexMatch Matchmaker erstellen, empfehlen wir, die Regelsatzsyntax zu überprüfen. Nachdem Sie einen Regelsatz mit der Amazon- GameLift Konsole oder der AWS Command Line Interface (AWS CLI) erstellt haben, können Sie ihn nicht mehr ändern.

Beachten Sie, dass es ein Servicekontingent für die maximale Anzahl von Regelsätzen gibt, die Sie in einer -AWSRegion haben können. Daher empfiehlt es sich, ungenutzte Regelsätze zu löschen.

Verwandte Themen

- Entwerfen Sie einen FlexMatch Regelsatz
- FlexMatch Beispiele für Regelsätze
- FlexMatchRegeln, Sprache

Console

Erstellen eines Regelsatzes

1. Öffnen Sie die Amazon- GameLift Konsole unter https://console.aws.amazon.com/gamelift/.
2. Wechseln Sie zu der AWS Region, in der Sie Ihren Regelsatz erstellen möchten. Definieren Sie Regelsätze in derselben Region wie die Matchmaking-Konfiguration, die sie verwendet.

3. Wählen Sie im Navigationsbereich , FlexMatch Matchmaking-Regelsätze aus.
4. Wählen Sie auf der Seite Matchmaking-Regelsätze die Option Regelsatz erstellen aus.
5. Gehen Sie auf der Seite Matchmaking-Regelsatz erstellen wie folgt vor:

   a. Geben Sie unter Regelsatzeinstellungen für Name einen eindeutigen beschreibenden Namen ein, mit dem Sie ihn in einer Liste oder in Ereignis- und Metriktabellen identifizieren können.

   b. Geben Sie für Regelsatz Ihren Regelsatz in JSON ein. Informationen zum Entwerfen eines Regelsatzes finden Sie unter Entwerfen Sie einen FlexMatch Regelsatz. Sie können auch einen der Beispielregelsätze von verwendenFlexMatch Beispiele für Regelsätze.

   c. Wählen Sie Validieren, um zu überprüfen, ob die Syntax Ihres Regelsatzes korrekt ist. Sie können Regelsätze nicht bearbeiten, nachdem sie erstellt wurden. Daher empfiehlt es sich, sie zuerst zu validieren.

   d. (Optional) Fügen Sie unter Tags Tags hinzu, um Sie bei der Verwaltung und Nachverfolgung Ihrer -AWSRessourcen zu unterstützen.
6. Wählen Sie Erstellen. Wenn die Erstellung erfolgreich ist, können Sie den Regelsatz mit einem Matchmaker verwenden.

AWS CLI

Erstellen eines Regelsatzes

Öffnen Sie ein Befehlszeilenfenster und verwenden Sie den Befehl create-matchmaking-rule-set.

Dieser Beispielbefehl erstellt einen einfachen Matchmaking-Regelsatz, der ein einzelnes Team einrichtet. Stellen Sie sicher, dass Sie den Regelsatz in derselben AWS Region wie die Matchmaking-Konfigurationen erstellen, die ihn verwenden.

```
aws gamelift create-matchmaking-rule-set \
    --name "SampleRuleSet123" \
    --rule-set-body '{"name": "aliens_vs_cowboys", "ruleLanguageVersion": "1.0",
 "teams": [{"name": "cowboys", "maxPlayers": 8, "minPlayers":  4}]}'
```

Wenn die Erstellungsanforderung erfolgreich ist, gibt Amazon ein MatchmakingRuleSet Objekt GameLift zurück, das die von Ihnen angegebenen Einstellungen enthält. Ein Matchmaker kann jetzt den neuen Regelsatz verwenden.

Console

Löschen eines Regelsatzes

1. Öffnen Sie die Amazon- GameLift Konsole unter https://console.aws.amazon.com/gamelift/.
2. Wechseln Sie zu der Region, in der Sie den Regelsatz erstellt haben.
3. Wählen Sie im Navigationsbereich , FlexMatch Matchmaking-Regelsätze aus.
4. Wählen Sie auf der Seite Matchmaking-Regelsätze den Regelsatz aus, den Sie löschen möchten, und wählen Sie dann Löschen aus.
5. Wählen Sie im Dialogfeld Regelsatz löschen die Option Löschen aus, um das Löschen zu bestätigen.

> **Note**
>
> Wenn eine Matchmaking-Konfiguration den Regelsatz verwendet, zeigt Amazon GameLift eine Fehlermeldung an (Regelsatz kann nicht gelöscht werden). Ändern Sie in diesem Fall die Matchmaking-Konfiguration, um einen anderen Regelsatz zu verwenden, und versuchen Sie es dann erneut. Um herauszufinden, welche Matchmaking-Konfigurationen einen Regelsatz verwenden, wählen Sie den Namen eines Regelsatzes aus, um seine Detailseite anzuzeigen.

AWS CLI

Löschen eines Regelsatzes

Öffnen Sie ein Befehlszeilenfenster und verwenden Sie den Befehl delete-matchmaking-rule-set, um einen Matchmaking-Regelsatz zu löschen.

Wenn eine Matchmaking-Konfiguration den Regelsatz verwendet, gibt Amazon eine Fehlermeldung GameLift zurück. Ändern Sie in diesem Fall die Matchmaking-Konfiguration, um einen anderen Regelsatz zu verwenden, und versuchen Sie es dann erneut. Um eine Liste der Matchmaking-Konfigurationen abzurufen, die einen Regelsatz verwenden, verwenden Sie den Befehl describe-matchmaking-configurations und geben Sie den Regelsatznamen an.

Dieser Beispielbefehl prüft die Verwendung des Matchmaking-Regelsatzes und löscht dann den Regelsatz.

```
aws gamelift describe-matchmaking-rule-sets \
    --rule-set-name "SampleRuleSet123" \
    --limit 10

aws gamelift delete-matchmaking-rule-set \
    --name  "SampleRuleSet123"
```

FlexMatch Beispiele für Regelsätze

FlexMatch -Regelsätze können eine Vielzahl von Matchmaking-Szenarien abdecken. Die folgenden Beispiele entsprechen der FlexMatch Konfigurationsstruktur und der Sprache des Eigenschaftsausdrucks. Kopieren Sie diesen Regelsatz komplett, oder wählen Sie nach Bedarf Komponenten davon aus.

Weitere Informationen zur Verwendung von FlexMatch Regeln und Regelsätzen finden Sie in den folgenden Themen:

- Erstellen Sie einen FlexMatch Regelsatz
- Entwerfen Sie einen FlexMatch Regelsatz
- FlexMatchRegelsatzschema
- FlexMatchRegeln, Sprache

> **Note**
>
> Bei der Auswertung eines Matchmaking-Tickets mit mehreren Spielern müssen alle Spieler in der Anforderungen die Match-Anforderungen erfüllen.

Beispiel 1: Erstellen von zwei Teams mit gleichmäßig übereinstimmenden Spielern

In diesem Beispiel wird gezeigt, wie Sie zwei gleichmäßig abgeglichenen Spielerteams mit den folgenden Anweisungen erstellen.

- Erstellen Sie zwei Spielerteams.
  - Nehmen Sie zwischen vier und acht Spieler in jedes Team auf.
  - Fertige Teams müssen die gleiche Anzahl von Spielern haben.

- Berücksichtigen Sie die Qualifikationsstufe eines Spielers (falls nicht vorhanden, standardmäßig 10).
- Wählen Sie Spieler abhängig davon aus, ob ihre Qualifikation ähnlich der der anderen Spieler ist. Stellen Sie sicher, dass beide Teams durchschnittliche Spielerqualifikationen innerhalb einer Toleranz von 10 Punkten zueinander aufweisen.
- Wenn das Match nicht schnell gefüllt wird, lockern Sie die Qualifikationsanforderung, um innerhalb einer angemessenen Zeit ein Match zu erstellen.
  - Nach 5 Sekunden erweitern Sie die Suche so, dass Teams mit durchschnittlichen Spielerqualifikationen in einem Bereich von 50 Punkten zulässig sind.
  - Nach 15 Sekunden erweitern Sie die Suche so, dass Teams mit durchschnittlichen Spielerqualifikationen in einem Bereich von 100 Punkten zulässig sind.

Hinweise zur Verwendung dieses Regelsatzes:

- Dieses Beispiel lässt Teams einer beliebigen Größe zwischen vier und acht Spielern zu (obwohl sie dieselbe Größe haben müssen). Für Teams mit mehreren gültigen Größen, versucht der Matchmaker, die maximale Anzahl zulässiger Spieler so gut wie möglich zu erfüllen.
- Die `FairTeamSkill`-Regel stellt sicher, dass die Teams basierend auf den Spielerqualifikationen gleichmäßig abgeglichen sind. Um diese Regel für jeden neuen potenziellen Spieler auszuwerten, fügt FlexMatch den Spieler vorläufig einem Team hinzu und berechnet die Durchschnittswerte. Wenn eine Regel fehlschlägt, wird der potenzielle Spieler dem Match nicht hinzugefügt.
- Da die Strukturen beider Teams identisch sind, könnten Sie nur eine Teamdefinition erstellen und als Teamanzahl „2“ festlegen. Wenn Sie dem Team in diesem Szenario den Namen „aliens“ geben würden, dann würden Ihren Teams die Namen „aliens_1“ und „aliens_2“. zugewiesen werden.

```
{
    "name": "aliens_vs_cowboys",
    "ruleLanguageVersion": "1.0",
    "playerAttributes": [{
        "name": "skill",
        "type": "number",
        "default": 10
    }],
    "teams": [{
        "name": "cowboys",
        "maxPlayers": 8,
```

```
        "minPlayers": 4
    }, {
        "name": "aliens",
        "maxPlayers": 8,
        "minPlayers": 4
    }],
    "rules": [{
        "name": "FairTeamSkill",
        "description": "The average skill of players in each team is within 10 points
 from the average skill of all players in the match",
        "type": "distance",
        // get skill values for players in each team and average separately to produce
 list of two numbers
        "measurements": [ "avg(teams[*].players.attributes[skill])" ],
        // get skill values for players in each team, flatten into a single list, and
 average to produce an overall average
        "referenceValue": "avg(flatten(teams[*].players.attributes[skill]))",
        "maxDistance": 10 // minDistance would achieve the opposite result
    }, {
        "name": "EqualTeamSizes",
        "description": "Only launch a game when the number of players in each team
 matches, e.g. 4v4, 5v5, 6v6, 7v7, 8v8",
        "type": "comparison",
        "measurements": [ "count(teams[cowboys].players)" ],
        "referenceValue": "count(teams[aliens].players)",
        "operation": "=" // other operations: !=, <, <=, >, >=
    }],
    "expansions": [{
        "target": "rules[FairTeamSkill].maxDistance",
        "steps": [{
            "waitTimeSeconds": 5,
            "value": 50
        }, {
            "waitTimeSeconds": 15,
            "value": 100
        }]
    }]
}
```

Beispiel 2: Erstellen ungleichmäßiger Teams (Hunters vs. Monster)

Dieses Beispiel beschreibt einen Spielmodus, wobei eine Gruppe von Spielern ein einziges Monster jagt. Die Spieler wählen Sie die Jäger- oder die Monster-Rolle. Jäger geben das

Mindestqualifikationsniveau für das Monster an, das sie jagen wollen. Die Mindestgröße des Jägerteams kann im Laufe der Zeit gelockert werden, um das Match fertigzustellen. Dieses Szenario enthält die folgenden Anweisungen:

- Erstellen Sie ein Team von genau fünf Jägern.
- Erstellen Sie ein separates Team mit genau einem Monster.
- Berücksichtigen Sie die folgenden Spielerattribute:
  - Die Qualifikationsstufe eines Spielers (falls nicht vorhanden, standardmäßig 10).
  - Die bevorzugte Monster-Qualifikationsstufe eines Spielers (falls nicht vorhanden, standardmäßig 10).
  - Ob der Spieler das Monster sein will (falls nicht vorhanden, standardmäßig 0 oder false).
- Wählen Sie einen Spieler als Monster basierend auf den folgenden Kriterien:
  - Der Spieler muss die Monsterrolle angefordert haben.
  - Der Spieler muss das höchste Qualifikationsniveau haben, das die Spieler, die bereits im Jägerteam vorhanden sind, bevorzugen, oder dieses übertreffen.
- Wählen Sie Spieler für das Jägerteam basierend auf den folgenden Kriterien:
  - Spieler, die eine Monster-Rolle anfordern, können nicht dem Jägerteam beitreten.
  - Wenn die Monster-Rolle bereits belegt ist, muss der Spieler ein gewünschtes Monster-Qualifikationsniveau anfordern, das niedriger als die Qualifikation des vorgeschlagenen Monsters ist.
- Wird ein Match nicht schnell gefüllt, lockern Sie die Mindestgröße des Jägerteams wie folgt:
  - Nach 30 Sekunden darf ein Spiel mit nur vier Spielern im Jägerteam starten.
  - Nach 60 Sekunden darf ein Spiel mit nur drei Spielern im Jägerteam starten.

Hinweise zur Verwendung dieses Regelsatzes:

- Durch die Verwendung von zwei separaten Teams für Jäger und Monster können Sie die Mitgliedschaft basierend auf verschiedenen Kriterien bewerten.

```
{
    "name": "players_vs_monster_5_vs_1",
    "ruleLanguageVersion": "1.0",
    "playerAttributes": [{
        "name": "skill",
```

```
        "type": "number",
        "default": 10
    },{
        "name": "desiredSkillOfMonster",
        "type": "number",
        "default": 10
    },{
        "name": "wantsToBeMonster",
        "type": "number",
        "default": 0
    }],
    "teams": [{
        "name": "players",
        "maxPlayers": 5,
        "minPlayers": 5
    }, {
        "name": "monster",
        "maxPlayers": 1,
        "minPlayers": 1
    }],
    "rules": [{
        "name": "MonsterSelection",
        "description": "Only users that request playing as monster are assigned to the
monster team",
        "type": "comparison",
        "measurements": ["teams[monster].players.attributes[wantsToBeMonster]"],
        "referenceValue": 1,
        "operation": "="
    },{
        "name": "PlayerSelection",
        "description": "Do not place people who want to be monsters in the players
team",
        "type": "comparison",
        "measurements": ["teams[players].players.attributes[wantsToBeMonster]"],
        "referenceValue": 0,
        "operation": "="
    },{
        "name": "MonsterSkill",
        "description": "Monsters must meet the skill requested by all players",
        "type": "comparison",
        "measurements": ["avg(teams[monster].players.attributes[skill])"],
        "referenceValue":
"max(teams[players].players.attributes[desiredSkillOfMonster])",
        "operation": ">="
```

```
    }],
    "expansions": [{
        "target": "teams[players].minPlayers",
        "steps": [{
            "waitTimeSeconds": 30,
            "value": 4
        },{
            "waitTimeSeconds": 60,
            "value": 3
        }]
    }]
}
```

Beispiel 3: Festlegen von Anforderungen und Latenzlimits auf Teamebene

In diesem Beispiel wird veranschaulicht, wie Spielerteams eingerichtet und ein Regelsatz anstatt auf einzelne Spieler auf jedes Team angewandt wird. Anhand einer einzigen Definition werden drei gut abgestimmte Teams erstellt. Außerdem wird eine maximale Latenz für alle Spieler festgelegt. Maximalwerte für die Latzen können im Laufe der Zeit gelockert werden, um das Match zu vervollständigen. In diesem Beispiel werden die folgenden Anweisungen beschrieben:

- Erstellen Sie drei Spielerteams.
  - Nehmen Sie zwischen drei und fünf Spieler in jedes Team auf.
  - Fertige Teams müssen die gleiche oder fast die gleiche Anzahl von Spielern (plus/minus einem Spieler) aufweisen.
- Berücksichtigen Sie die folgenden Spielerattribute:
  - Die Qualifikationsstufe eines Spielers (falls nicht vorhanden, standardmäßig 10).
  - Die Charakterrolle eines Spielers (falls nicht vorhanden, standardmäßig „Bauer“).
- Wählen Sie Spieler abhängig davon aus, ob ihre Qualifikation ähnlich der der anderen Spieler im Match sind.
  - Stellen Sie sicher, dass beide Teams durchschnittliche Spielerqualifikationen innerhalb einer Toleranz von 10 Punkten zueinander aufweisen.
- Beschränken Sie Teams an die folgende Anzahl von „Heiler“-Charakteren:
  - Ein ganzes Match kann maximal fünf Heiler haben.
- Nehmen Sie nur Spieler in ein Match auf, die eine Latenz von 50 Millisekunden oder weniger verzeichnen.
- Wird ein Match nicht schnell gefüllt, lockern Sie die Anforderung an die Spieler-Latenz wie folgt:

- Nach 10 Sekunden lassen Sie Latenzwerte von bis zu 100 ms für die Spieler zu.
- Nach 20 Sekunden lassen Sie Latenzwerte von bis zu 150 ms für die Spieler zu.

Hinweise zur Verwendung dieses Regelsatzes:

- Der Regelsatz stellt sicher, dass die Teams basierend auf den Spielerqualifikationen gleichmäßig abgeglichen sind. Um die `FairTeamSkill` Regel zu bewerten, fügt den potenziellen Spieler FlexMatch einem Team hinzu und berechnet die durchschnittliche Fähigkeit der Spieler im Team. Anschließend wird die Regel mit der durchschnittlichen Qualifikation der Spieler in beiden Teams verglichen. Wenn eine Regel fehlschlägt, wird der potenzielle Spieler dem Match nicht hinzugefügt.
- Die Team- und Match-Level-Anforderungen (Gesamtanzahl der Heiler) werden durch eine Sammlungsregel erzielt. Dieser Regeltyp vergleicht eine Liste von Charakterattributen aller Spieler und verglicht sie mit der maximal zulässigen Anzahl. Verwenden Sie `flatten` zum Erstellen einer Liste aller Spieler in allen Teams.
- Bei der Auswertung von basierend auf Latenz, beachten Sie Folgendes:
  - Latenzdaten werden in der Matchmaking-Anforderung als Teil des Player-Objekts bereitgestellt. Es handelt sich nicht um ein Spielerattribut, sodass es nicht als solches aufgeführt werden muss.
  - Der Matchmaker bewertet die Latenz nach Region. Jede Region mit einer Latenz höher als der maximal zulässigen Latzen wird ignoriert. Um für ein Match akzeptiert zu werden, muss ein Spieler mindestens eine Region mit einer Latenz unterhalb der maximal zulässigen Latenz haben.
  - Wenn Matchmaking-Anforderungen Latenzdaten für einen oder mehrere Spieler weglassen, wird die Anforderung für alle Matches abgelehnt.

```
{
    "name": "three_team_game",
    "ruleLanguageVersion": "1.0",
    "playerAttributes": [{
        "name": "skill",
        "type": "number",
        "default": 10
    },{
        "name": "character",
        "type": "string_list",
        "default": [ "peasant" ]
    }],
```

```
   "teams": [{
       "name": "trio",
       "minPlayers": 3,
       "maxPlayers": 5,
       "quantity": 3
   }],
   "rules": [{
       "name": "FairTeamSkill",
       "description": "The average skill of players in each team is within 10 points
from the average skill of players in the match",
       "type": "distance",
       // get players for each team, and average separately to produce list of 3
       "measurements": [ "avg(teams[*].players.attributes[skill])" ],
       // get players for each team, flatten into a single list, and average to
produce overall average
       "referenceValue": "avg(flatten(teams[*].players.attributes[skill]))",
       "maxDistance": 10 // minDistance would achieve the opposite result
   }, {
       "name": "CloseTeamSizes",
       "description": "Only launch a game when the team sizes are within 1 of each
other.  e.g. 3 v 3 v 4 is okay, but not 3 v 5 v 5",
       "type": "distance",
       "measurements": [ "max(count(teams[*].players))"],
       "referenceValue": "min(count(teams[*].players))",
       "maxDistance": 1
   }, {
       "name": "OverallMedicLimit",
       "description": "Don't allow more than 5 medics in the game",
       "type": "collection",
       // This is similar to above, but the flatten flattens everything into a single
       // list of characters in the game.
       "measurements": [ "flatten(teams[*].players.attributes[character])"],
       "operation": "contains",
       "referenceValue": "medic",
       "maxCount": 5
   }, {
       "name": "FastConnection",
       "description": "Prefer matches with fast player connections first",
       "type": "latency",
       "maxLatency": 50
   }],
   "expansions": [{
       "target": "rules[FastConnection].maxLatency",
       "steps": [{
```

```
            "waitTimeSeconds": 10,
            "value": 100
        }, {
            "waitTimeSeconds": 20,
            "value": 150
        }]
    }]
}
```

Beispiel 4: Verwenden Sie die explizite Sortierung, um die besten Übereinstimmungen zu finden

In diesem Beispiel wird ein einfaches Match mit zwei Teams mit jeweils drei Spielern eingerichtet. Es veranschaulicht, wie Sie explizite Sortierregeln anwenden, um die bestmöglichen Matches so schnell wie möglich zu finden. Diese Regeln sortieren alle aktiven Matchmaking-Tickets, um die besten Übereinstimmungen auf der Grundlage bestimmter Schlüsselanforderungen zu erstellen. Dieses Beispiel wird mit den folgenden Anweisungen implementiert:

- Erstellen Sie zwei Spielerteams.
- Nehmen Sie genau drei Spieler in jedes Team auf.
- Berücksichtigen Sie die folgenden Spielerattribute:
  - Qualifikationsniveau (falls nicht vorhanden, standardmäßig 50).
  - Bevorzugte Spielmodi (es können mehrere Werte angegeben werden) (falls nicht vorhanden, standardmäßig „coop“ und „deathmatch“).
  - Bevorzugte Spiel-Karten, einschließlich Kartenname und Prioritätsgewichtung (falls nicht vorhanden, standardmäßig `"defaultMap"`, mit einem Gewicht von 100).
- Einrichtung der Vorsortierung:
  - Sortieren Sie Spieler basierend auf ihrer Präferenz für dieselbe Spiel-Karte wie der Anker-Spieler. Spieler können mehrere bevorzugte Spiel-Karten haben, deshalb verwendet dieses Beispiel einen Präferenzwert.
  - Sortieren Sie Spieler anhand dessen, wie gut ihre Erfahrung mit der eines Anker-Spielers übereinstimmt. Dank dieser Sortierung verfügen alle Spieler in allen Teams über Erfahrungsniveaus, die so ähnlich wie möglich sind.
- Alle Spieler in allen Teams muss mindestens einen gemeinsamen Spiel-Modus ausgewählt haben.
- Alle Spieler in allen Teams muss mindestens eine gemeinsame Spiel-Karte ausgewählt haben.

Hinweise zur Verwendung dieses Regelsatzes:

- Die Sortierung nach Spiel-Karte verwendet eine absolute Sortierung, die den mapPreference-Attributwert vergleicht. Da diese Regel an erster Stelle des Regelsatzes steht, erfolgt diese Sortierung zuerst.
- Die Sortierung nach Erfahrung verwendet eine Distanzsortierung, um die Qualifikationsstufe eines potenziellen Spielers mit der Qualifikation des Anker-Spielers zu vergleichen.
- Sortierungen werden in der Reihenfolge ausgeführt, in der sie im Regelsatz aufgelistet werden. In diesem Szenario werden Spieler nach der Präferenz für ihre Spiel-Karte und dann nach dem Erfahrungswert sortiert.

```
{
    "name": "multi_map_game",
    "ruleLanguageVersion": "1.0",
    "playerAttributes": [{
        "name": "experience",
        "type": "number",
        "default": 50
    }, {
        "name": "gameMode",
        "type": "string_list",
        "default": [ "deathmatch", "coop" ]
    }, {
        "name": "mapPreference",
        "type": "string_number_map",
        "default": { "defaultMap": 100 }
    }, {
        "name": "acceptableMaps",
        "type": "string_list",
        "default": [ "defaultMap" ]
    }],
    "teams": [{
        "name": "red",
        "maxPlayers": 3,
        "minPlayers": 3
    }, {
        "name": "blue",
        "maxPlayers": 3,
        "minPlayers": 3
    }],
    "rules": [{
```

```
        // We placed this rule first since we want to prioritize players preferring the
 same map
        "name": "MapPreference",
        "description": "Favor grouping players that have the highest map preference
 aligned with the anchor's favorite",
        // This rule is just for sorting potential matches.  We sort by the absolute
 value of a field.
        "type": "absoluteSort",
        // Highest values go first
        "sortDirection": "descending",
        // Sort is based on the mapPreference attribute.
        "sortAttribute": "mapPreference",
        // We find the key in the anchor's mapPreference attribute that has the highest
 value.
        // That's the key that we use for all players when sorting.
        "mapKey": "maxValue"
    }, {
        // This rule is second because any tie-breakers should be ordered by similar
 experience values
        "name": "ExperienceAffinity",
        "description": "Favor players with similar experience",
        // This rule is just for sorting potential matches.  We sort by the distance
 from the anchor.
        "type": "distanceSort",
        // Lowest distance goes first
        "sortDirection": "ascending",
        "sortAttribute": "experience"
    }, {
        "name": "SharedMode",
        "description": "The players must have at least one game mode in common",
        "type": "collection",
        "operation": "intersection",
        "measurements": [ "flatten(teams[*].players.attributes[gameMode])"],
        "minCount": 1
    }, {
        "name": "MapOverlap",
        "description": "The players must have at least one map in common",
        "type": "collection",
        "operation": "intersection",
        "measurements": [ "flatten(teams[*].players.attributes[acceptableMaps])"],
        "minCount": 1
    }]
}
```

Beispiel 5: Bestimmung von Schnittmengen über mehrere Spielerattribute hinweg

Dieses Beispiel veranschaulicht, wie Sie mit einer Sammlungsregel Schnittmengen in zwei oder mehreren Spielerattributen finden. Beim Arbeiten mit Sammlungen können Sie die `intersection`-Operation für ein einzelnes Attribut und die `reference_intersection_count`-Operation für mehrere Attribute verwenden.

Um diesen Ansatz zu verdeutlichen, wertet dieses Beispiel Spieler in einem Match basierend auf ihren Charakter-Präferenzen aus. Das Beispielspiel ist ein „free-for-all“-Stil, bei dem alle Spieler in einem Spiel Opponenten sind. Jeder Spieler wird aufgefordert, (1) einen Charakter für sich zu wählen, und (2) Charaktere zu wählen, gegen die er spielen möchte. Wir brauchen eine Regel, die gewährleistet, dass jeder Spieler in einem Match einen Charakter verwendet, der auf der Liste bevorzugter Gegner der anderen Spieler steht.

Der Beispielregelsatz beschreibt ein Match mit den folgenden Eigenschaften:

- Team-Struktur: Ein Team mit fünf Spielern
- Spielerattribute:
  - myCharacter: Der vom Spieler ausgewählte Charakter.
  - preferredOpponents: Liste der Charaktere, gegen die der Spieler spielen will.
- Match-Regeln: Ein potenzielles Match ist akzeptabel, wenn sich jeder der verwendeten Charaktere auf der Liste der bevorzugten Gegner aller Spieler befindet.

Um die Match-Regeln zu implementieren, verwendet dieses Beispiel eine Sammlungsregel mit den folgenden Eigenschaftswerten:

- Operation – Verwendet die -`reference_intersection_count`Operation, um auszuwerten, wie sich jede Zeichenfolgenliste im Messungswert mit der Zeichenfolgenliste im Referenzwert schneidet.
- Messung – Verwendet den `flatten` Eigenschaftsausdruck, um eine Liste von Zeichenfolgenlisten zu erstellen, wobei jede Zeichenfolgenliste den myCharacter-Attributwert eines Spielers enthält.
- Referenzwert – Verwendet den -`set_intersection`Eigenschaftsausdruck, um eine Zeichenfolgenliste aller preferredOpponents-Attributwerte zu erstellen, die jedem Spieler im Match gemeinsam sind.
- Einschränkungen – `minCount` ist auf 1 gesetzt, um sicherzustellen, dass das von jedem Spieler gewählte Zeichen (eine Zeichenfolgenliste in der Messung) mit mindestens einem der

bevorzugten Opponenten übereinstimmt, die allen Spielern gemeinsam sind. (eine Zeichenfolge im Referenzwert).

- Erweiterung – Wenn eine Übereinstimmung nicht innerhalb von 15 Sekunden erfüllt wird, reduzieren Sie die Mindestüberschneidungsanforderung.

Der Verarbeitungsablauf für diese Regel sieht wie folgt aus:

1. Ein Spieler wird dem potenziellen Match hinzugefügt. Der Referenzwert (eine Zeichenfolgenliste) wird neu berechnet, um die Schnittmengen mit der Liste der bevorzugten Gegner des neuen Spielers zu beinhalten. Der Messwert (eine Liste von Zeichenfolgenlisten) wird neu berechnet, um den von dem neuen Spieler gewählten Charakter als neue Zeichenfolgenliste hinzuzufügen.
2. Amazon GameLift überprüft, ob sich jede Zeichenfolgenliste im Messungswert (die von den Spielern ausgewählten Zeichen) mit mindestens einer Zeichenfolge im Referenzwert (die bevorzugten Opponenten der Spieler) schneidet. Da in diesem Beispiel jede Zeichenfolgenliste in der Messung nur einen Wert enthält, ist die Schnittmenge entweder 0 oder 1.
3. Wenn eine Zeichenfolgenliste in der Messung keine Schnittmenge mit der Referenzwert-Zeichenfolgenliste hat, schlägt die Regel fehl und der neue Spieler wird aus dem potenziellen Match entfernt.
4. Wird ein Match nicht innerhalb von 15 Sekunden gefüllt, wird die Match-Anforderung des Gegners verworfen, die restlichen Spielerplätze im Match zu füllen.

```
{
    "name": "preferred_characters",
    "ruleLanguageVersion": "1.0",

    "playerAttributes": [{
        "name": "myCharacter",
        "type": "string_list"
    }, {
        "name": "preferredOpponents",
        "type": "string_list"
    }],

    "teams": [{
        "name": "red",
        "minPlayers": 5,
        "maxPlayers": 5
    }],
```

```
    "rules": [{
        "description": "Make sure that all players in the match are using a character
 that is on all other players' preferred opponents list.",
        "name": "OpponentMatch",
        "type": "collection",
        "operation": "reference_intersection_count",
        "measurements": ["flatten(teams[*].players.attributes[myCharacter])"],
        "referenceValue":
 "set_intersection(flatten(teams[*].players.attributes[preferredOpponents]))",
        "minCount":1
    }],
    "expansions": [{
        "target": "rules[OpponentMatch].minCount",
        "steps": [{
            "waitTimeSeconds": 15,
            "value": 0
        }]
    }]
}
```

Beispiel 6: Vergleichen von Attributen in allen Spielern

Dieses Beispiel veranschaulicht, wie Spielerattribute innerhalb einer Gruppe von Spielern verglichen werden.

Der Beispielregelsatz beschreibt ein Match mit den folgenden Eigenschaften:

- Teamstruktur: Zwei Singleplayer-Teams
- Spielerattribute:
  - gameMode: Art des vom Spieler gewählten Spiels (falls nicht angegeben, wird standardmäßig „rundenbasiert“ verwendet).
  - gameMap: Vom Spieler gewählte Spielwelt (falls nicht anders angegeben, standardmäßig 1).
  - character: Vom Spieler gewählte Spielfigur (kein Standardwert bedeutet, dass der Spieler eine Spielfigur angeben muss).
- Match-Regeln: In einem Match platzierte Spieler müssen die folgenden Anforderungen erfüllen:
  - Die Spieler müssen denselben Spiel-Modus wählen.
  - Die Spieler müssen dieselbe Spielekarte wählen.
  - Die Spieler müssen unterschiedliche Charaktere wählen.

Hinweise zur Verwendung dieses Regelsatzes:

- Um die Match-Regel zu implementieren, verwendet dieses Beispiel Vergleichsregeln, um die Attributwerte aller Spieler zu vergleichen. Für den Spielmodus und die Karte überprüft die Regel, ob die Werte identisch sind. Für den Charakter überprüft die Regel, ob die Werte unterschiedlich sind.
- Dieses Beispiel verwendet eine Spieler-Definition mit einer Mengen-Eigenschaft zum Erstellen beider Spieler-Teams. Dem Team werden die folgenden Namen zugewiesen: „player_1“ und „player_2“.

```
{
    "name": "",
    "ruleLanguageVersion": "1.0",

    "playerAttributes": [{
        "name": "gameMode",
        "type": "string",
        "default": "turn-based"
    }, {
        "name": "gameMap",
        "type": "number",
        "default": 1
    }, {
        "name": "character",
        "type": "number"
    }],

    "teams": [{
        "name": "player",
        "minPlayers": 1,
        "maxPlayers": 1,
        "quantity": 2
    }],

    "rules": [{
        "name": "SameGameMode",
        "description": "Only match players when they choose the same game type",
        "type": "comparison",
        "operation": "=",
        "measurements": ["flatten(teams[*].players.attributes[gameMode])"]
    }, {
```

```
        "name": "SameGameMap",
        "description": "Only match players when they're in the same map",
        "type": "comparison",
        "operation": "=",
        "measurements": ["flatten(teams[*].players.attributes[gameMap])"]
    }, {
        "name": "DifferentCharacter",
        "description": "Only match players when they're using different characters",
        "type": "comparison",
        "operation": "!=",
        "measurements": ["flatten(teams[*].players.attributes[character])"]
    }]
}
```

Beispiel 7: Erstellen einer großen Übereinstimmung

In diesem Beispiel wird veranschaulicht, wie Sie einen Regelsatz für Matches mit mehr als 40 Spielern einrichten. Wenn ein Regelsatz Teams mit einer maxPlayer-Gesamtzahl größer als 40 beschreibt, wird er als großes Match verarbeitet. Weitere Informationen finden Sie unter Entwerfen Sie einen FlexMatch Regelsatz für große Übereinstimmungen.

Der Beispiel-Regelsatz erstellt ein Match unter Beachtung der folgenden Anweisungen:

- Erstellen Sie ein Team mit bis zu 200 Spielern mit einer Mindestanforderung von 175 Spielern.
- Ausgleichende Kriterien: Wählen Sie Spieler basierend auf vergleichbarer Qualifikationsstufe aus. Alle Spieler müssen ihre Qualifikationsstufe angeben, um in ein Match aufgenommen zu werden.
- Stapelverarbeitungs-Präferenz: Gruppieren Sie Spieler beim Erstellen von Matches nach ähnlichen ausgleichenden Kritieren.
- Latenzregeln: Legen Sie als maximal zulässige Spieler-Latenz 150 Millisekunden fest.
- Wenn das Match nicht schnell gefüllt wird, lockern Sie die Anforderung, um innerhalb einer angemessenen Zeit ein Match fertig zu stellen.
  - Akzeptieren Sie nach 10 Sekunden ein Team mit 150 Spielern.
  - Erhöhen Sie nach 12 Sekunden die maximale akzeptable Latenz auf 200 Millisekunden.
  - Akzeptieren Sie nach 15 Sekunden ein Team mit 100 Spielern.

Hinweise zur Verwendung dieses Regelsatzes:

- Da der Algorithmus die Stapelverarbeitungs-Präferenz „largestPopulation“ verwendet, werden Spieler zuerst basierend auf den ausgleichenden Kriterien sortiert. Dies hat zur Folge, dass Matches meist voller sind und Spieler mit ähnlicherer Qualifikation enthalten. Alle Spieler erfüllen akzeptable Latenzanforderungen, erhalten möglicherweise aber nicht die bestmögliche Latenz für ihren Ort.
- Die in diesem Regelsatz verwendete Algorithmusstrategie „largestPopulation“ ist die Standardeinstellung. Wenn Sie die Standardeinstellung verwenden möchten, müssen Sie die Einstellung nicht explizit angeben.
- Wenn Sie Match-Backfill aktiviert haben, lockern Sie die erforderliche Spieleranzahl nicht zu schnell. Andernfalls erhalten Sie zu viele nur teilweise gefüllte Spielsitzungen. Weitere Informationen finden Sie unter Relaxen Sie große Spielanforderungen.

```
{
    "name": "free-for-all",
    "ruleLanguageVersion": "1.0",
    "playerAttributes": [{
        "name": "skill",
        "type": "number"
    }],
    "algorithm": {
        "balancedAttribute": "skill",
        "strategy": "balanced",
        "batchingPreference": "largestPopulation"
    },
    "teams": [{
        "name": "Marauders",
        "maxPlayers": 200,
        "minPlayers": 175
    }],
    "rules": [{
        "name": "low-latency",
        "description": "Sets maximum acceptable latency",
        "type": "latency",
        "maxLatency": 150
    }],
    "expansions": [{
        "target": "rules[low-latency].maxLatency",
        "steps": [{
            "waitTimeSeconds": 12,
            "value": 200
```

```
        }],
    }, {
        "target": "teams[Marauders].minPlayers",
        "steps": [{
            "waitTimeSeconds": 10,
            "value": 150
        }, {
            "waitTimeSeconds": 15,
            "value": 100
        }]
    }]
}
```

Beispiel 8: Erstellen einer großen Übereinstimmung mit mehreren Teams

In diesem Beispiel wird veranschaulicht, wie ein Regelsatz für Matches mit mehreren Teams eingerichtet wird, die mehr als 40 Spieler enthalten können. Es wird aufgezeigt, wie mit einer Definition mehrere identische Teams erstellt werden und wie Teams asymmetrischer Größe bei der Erstellung des Matches gefüllt werden.

Der Beispiel-Regelsatz erstellt ein Match unter Beachtung der folgenden Anweisungen:

- Erstellen Sie zehn identische „Jäger“-Teams mit bis zu 15 Spielern und ein „Monster“-Team mit genau fünf Spielern.
- Ausgleichende Kriterien: Wählen Sie Spieler basierend auf der Anzahl der Monster-Kills aus. Verwenden Sie bei Spielern, für die keine Kill-Anzahl verzeichnet wird, als Standardwert 5.
- Stapelverarbeitungs-Präferenzen: Gruppieren Sie Spieler basierend auf den Regionen, in denen sie die schnellste Spieler-Latenz verzeichnen.
- Latenzregel: Legen Sie als maximal zulässige Spieler-Latenz 200 Millisekunden fest.
- Wenn das Match nicht schnell gefüllt wird, lockern Sie die Anforderung, um innerhalb einer angemessenen Zeit ein Match fertig zu stellen.
  - Akzeptieren Sie nach 15 Sekunden Teams mit 10 Spielern.
  - Akzeptieren Sie nach 20 Sekunden Teams mit 8 Spielern.

Hinweise zur Verwendung dieses Regelsatzes:

- Dieser Regelsatz definiert Teams, die potenziell bis zu 155 Spieler aufnehmen können, was es zu einem großen Match macht. (10 x 15 Spieler + 5 Monster = 155)

- Da der Algorithmus als Stapelverarbeitungs-Präferenz die „schnellste Region“ verwendet, werden Spieler verstärkt in Regionen mit schnellerer verzeichneter Latenz und nicht in Regionen mit hoher (aber akzeptabler) verzeichneter Latenz platziert. Gleichzeitig besitzen Matches wahrscheinlich weniger Spieler, und das ausgleichende Kriterium (Anzahl von Monster-Kills) kann stärker variieren.
- Wenn eine Erweiterung für eine Multi-Team-Definition (Menge >1) definiert ist, gilt die Erweiterung für alle Teams, die mit dieser Definition erstellt wurden. Von einer Lockerung der minimalen Einstellung der Spieler im Jäger-Team sind alle zehn Jäger Teams gleichermaßen betroffen.
- Da dieser Regelsatz zum Minimieren der Spieler-Latenz optimiert ist, fungiert die Latenz-Regel als Catch-all-Methode zum Ausschließen von Spielern ohne akzeptable Verbindungsoptionen. Wir müssen diese Anforderung nicht lockern.
- So FlexMatch füllt Übereinstimmungen für diesen Regelsatz aus, bevor Erweiterungen wirksam werden:
  - Keines der Teams hat die minPlayers-Anzahl erreicht. Jäger-Teams besitzen über 15 verfügbare Spielerplätze, während das Monster-Team 5 verfügbare Spielerplätze hat.
    - Die ersten 100 Spieler werden (jeweils 10) den zehn Jäger-Teams zugewiesen.
    - Die nächsten 22 Spielern werden sequenziell (jeweils 2) den Jäger-Teams und dem Moster-Team zugewiesen.
  - Jäger-Teams haben die minPlayers-Anzahl von jeweils 12 Spielern erreicht. Das Monster-Team besitzt 2 Spieler und hat die minPlayers-Anzahl noch nicht erreicht.
    - Die nächsten drei Spieler werden dem Monster-Team zugewiesen.
  - Alle Teams haben die minPlayers-Anzahl erreicht. Jäger-Teams besitzen jeweils drei verfügbare Spielerplätze. Das Monster-Team ist voll.
    - Die letzten 30 Spieler werden sequenziell den Jäger-Teams zugewiesen. Dadurch wird sichergestellt, dass alle Jäger-Teams in etwa (plus oder minus einem Spieler) die gleiche Größe aufweisen.
- Wenn Sie Backfill für die mit diesem Regelsatz erstellten Matches aktiviert haben, lockern Sie die erforderliche Spieleranzahl nicht zu schnell. Andernfalls erhalten Sie zu viele nur teilweise gefüllte Spielsitzungen. Weitere Informationen finden Sie unter Relaxen Sie große Spielanforderungen.

```
{
    "name": "monster-hunters",
    "ruleLanguageVersion": "1.0",
    "playerAttributes": [{
```

```
        "name": "monster-kills",
        "type": "number",
        "default": 5
    }],
    "algorithm": {
        "balancedAttribute": "monster-kills",
        "strategy": "balanced",
        "batchingPreference": "fastestRegion"
    },
    "teams": [{
        "name": "Monsters",
        "maxPlayers": 5,
        "minPlayers": 5
    }, {
        "name": "Hunters",
        "maxPlayers": 15,
        "minPlayers": 12,
        "quantity": 10
    }],
    "rules": [{
        "name": "latency-catchall",
        "description": "Sets maximum acceptable latency",
        "type": "latency",
        "maxLatency": 150
    }],
    "expansions": [{
        "target": "teams[Hunters].minPlayers",
        "steps": [{
            "waitTimeSeconds": 15,
            "value": 10
        }, {
            "waitTimeSeconds": 20,
            "value": 8
        }]
    }]
}
```

Beispiel 9: Erstellen eines großen Matches mit Spielern mit ähnlichen Attributen

Dieses Beispiel veranschaulicht, wie Sie mithilfe von einen Regelsatz für Matches mit zwei Teams einrichten`batchDistance`. Im Beispiel:

- Die `SimilarLeague` Regel stellt sicher, dass alle Spieler in einem Spiel `league` innerhalb von 2 anderen Spielern ein haben.
- Die `SimilarSkill` Regel stellt sicher, dass alle Spieler in einem Spiel `skill` innerhalb von 10 anderen Spielern ein haben. Wenn ein Spieler 10 Sekunden gewartet hat, wird die Entfernung auf 20 erweitert. Wenn ein Spieler 20 Sekunden gewartet hat, wird die Entfernung auf 40 erweitert.
- Die `SameMap` Regel stellt sicher, dass alle Spieler in einem Spiel denselben angefordert haben`map`.
- Die `SameMode` Regel stellt sicher, dass alle Spieler in einem Spiel denselben angefordert haben`mode`.

```
{
    "ruleLanguageVersion": "1.0",
    "teams": [{
        "name": "red",
        "minPlayers": 100,
        "maxPlayers": 100
    }, {
        "name": "blue",
        "minPlayers": 100,
        "maxPlayers": 100
    }],
    "algorithm": {
        "strategy":"balanced",
        "balancedAttribute": "skill",
        "batchingPreference":"fastestRegion"
    },
    "playerAttributes": [{
        "name": "league",
        "type": "number"
    },{
        "name": "skill",
        "type": "number"
    },{
        "name": "map",
        "type": "string"
    },{
        "name": "mode",
        "type": "string"
    }],
    "rules": [{
        "name": "SimilarLeague",
```

```
        "type": "batchDistance",
        "batchAttribute": "league",
        "maxDistance": 2
    }, {
        "name": "SimilarSkill",
        "type": "batchDistance",
        "batchAttribute": "skill",
        "maxDistance": 10
    }, {
        "name": "SameMap",
        "type": "batchDistance",
        "batchAttribute": "map"
    }, {
        "name": "SameMode",
        "type": "batchDistance",
        "batchAttribute": "mode"
    }],
    "expansions": [{
        "target": "rules[SimilarSkill].maxDistance",
        "steps": [{
            "waitTimeSeconds": 10,
            "value": 20
        }, {
            "waitTimeSeconds": 20,
            "value": 40
        }]
    }]
}
```

Beispiel 10: Verwenden Sie eine zusammengesetzte Regel, um ein Match mit Spielern mit ähnlichen Attributen oder ähnlichen Auswahlen zu erstellen

Dieses Beispiel veranschaulicht, wie Sie mithilfe von einen Regelsatz für Matches mit zwei Teams einrichten`compound`. Im Beispiel:

- Die `SimilarLeagueDistance` Regel stellt sicher, dass alle Spieler in einem Spiel `league` innerhalb von 2 anderen Spielern ein haben.
- Die `SimilarSkillDistance` Regel stellt sicher, dass alle Spieler in einem Spiel `skill` innerhalb von 10 anderen Spielern ein haben. Wenn ein Spieler 10 Sekunden gewartet hat, wird die Entfernung auf 20 erweitert. Wenn ein Spieler 20 Sekunden gewartet hat, wird die Entfernung auf 40 erweitert.

- Die `SameMapComparison` Regel stellt sicher, dass alle Spieler in einem Spiel denselben angefordert habenmap.
- Die `SameModeComparison` Regel stellt sicher, dass alle Spieler in einem Spiel denselben angefordert habenmode.
- Die `CompoundRuleMatchmaker` Regel stellt eine Übereinstimmung sicher, wenn mindestens eine der folgenden Bedingungen erfüllt ist:
  - Spieler in einem Match haben dieselbe `map` und dieselbe angefordertmode.
  - Spieler in einer Übereinstimmung haben vergleichbare - `skill` und -`league`Attribute.

```
{
    "ruleLanguageVersion": "1.0",
    "teams": [{
        "name": "red",
        "minPlayers": 10,
        "maxPlayers": 20
    }, {
        "name": "blue",
        "minPlayers": 10,
        "maxPlayers": 20
    }],
    "algorithm": {
        "strategy":"balanced",
        "balancedAttribute": "skill",
        "batchingPreference":"fastestRegion"
    },
    "playerAttributes": [{
        "name": "league",
        "type": "number"
    },{
        "name": "skill",
        "type": "number"
    },{
        "name": "map",
        "type": "string"
    },{
        "name": "mode",
        "type": "string"
    }],
    "rules": [{
        "name": "SimilarLeagueDistance",
```

```
        "type": "distance",
        "measurements": ["max(flatten(teams[*].players.attributes[league]))"],
        "referenceValue": "min(flatten(teams[*].players.attributes[league]))",
        "maxDistance": 2
    }, {
        "name": "SimilarSkillDistance",
        "type": "distance",
        "measurements": ["max(flatten(teams[*].players.attributes[skill]))"],
        "referenceValue": "min(flatten(teams[*].players.attributes[skill]))",
        "maxDistance": 10
    }, {
        "name": "SameMapComparison",
        "type": "comparison",
        "operation": "=",
        "measurements": ["flatten(teams[*].players.attributes[map])"]
    }, {
        "name": "SameModeComparison",
        "type": "comparison",
        "operation": "=",
        "measurements": ["flatten(teams[*].players.attributes[mode])"]
    }, {
        "name": "CompoundRuleMatchmaker",
        "type": "compound",
        "statement": "or(and(SameMapComparison, SameModeComparison),
 and(SimilarSkillDistance, SimilarLeagueDistance))"
    }],
    "expansions": [{
        "target": "rules[SimilarSkillDistance].maxDistance",
        "steps": [{
            "waitTimeSeconds": 10,
            "value": 20
        }, {
            "waitTimeSeconds": 20,
            "value": 40
        }]
    }]
}
```

Beispiel 11: Erstellen einer Regel, die die Blockliste eines Spielers verwendet

Dieses Beispiel zeigt einen Regelsatz, mit dem Spieler vermeiden können, mit bestimmten anderen Spielern abgeglichen zu werden. Spieler können eine Blockliste erstellen, die der Matchmaker bei der

Spielerauswahl für ein Match auswertet. Weitere Hinweise zum Hinzufügen einer Blockliste oder zum Vermeiden einer Listenfunktion finden Sie unter AWS für Games Blog.

In diesem Beispiel werden die folgenden Anweisungen beschrieben:

- Erstellen Sie zwei Teams mit genau fünf Spielern.
- Übergeben Sie die Blockierliste eines Spielers, bei der es sich um eine Liste von Spieler-IDs (bis zu 100) handelt.
- Vergleichen Sie alle Spieler mit der Blockliste jedes Spielers und lehnen Sie ein vorgeschlagenes Match ab, wenn blockierte Spieler-IDs gefunden werden.

Hinweise zur Verwendung dieses Regelsatzes:

- Wenn Sie einen neuen Spieler auswerten, um ihn zu einem vorgeschlagenen Match hinzuzufügen (oder einen Spot in einem vorhandenen Match aufzufüllen), kann der Spieler aus einem der folgenden Gründe abgelehnt werden:
  - Wenn sich der neue Spieler auf einer Blockliste für Spieler befindet, die bereits für das Spiel ausgewählt sind.
  - Wenn Spieler, die bereits für das Spiel ausgewählt sind, auf der Blockierliste des neuen Spielers stehen.
- Wie gezeigt, verhindert dieser Regelsatz, dass ein Spieler mit einem Spieler auf seiner Blockliste übereinstimmt. Sie können diese Anforderung in eine Präferenz ändern (auch als „Vermeidungs“-Liste bezeichnet), indem Sie eine Regelerweiterung hinzufügen und den `maxCount` Wert erhöhen.

```
{
    "name": "Player Block List",
    "ruleLanguageVersion": "1.0",
    "teams": [{
        "maxPlayers": 5,
        "minPlayers": 5,
        "name": "red"
    }, {
        "maxPlayers": 5,
        "minPlayers": 5,
        "name": "blue"
    }],
    "playerAttributes": [{
        "name": "BlockList",
```

```
        "type": "string_list",
        "default": []
    }],
    "rules": [{
        "name": "PlayerIdNotInBlockList",
        "type": "collection",
        "operation": "reference_intersection_count",
        "measurements": "flatten(teams[*].players.attributes[BlockList])",
        "referenceValue": "flatten(teams[*].players[playerId])",
        "maxCount": 0
    }]
}
```

Erstellen Sie eine Matchmaking-Konfiguration

Um einen GameLift FlexMatch Amazon-Matchmaker für die Bearbeitung von Matchmaking-Anfragen einzurichten, erstellen Sie eine Matchmaking-Konfiguration. Verwenden Sie entweder die GameLift Amazon-Konsole oder die AWS Command Line Interface (AWS CLI). Weitere Informationen zum Erstellen eines Matchmakers finden Sie unterEntwerfen eines FlexMatch Matchmakers.

Erstellen Sie einen Matchmaker für Amazon-Hosting GameLift

Bevor Sie eine Matchmaking-Konfiguration erstellen, erstellen Sie einen Regelsatz und eine Warteschlange für GameLift Amazon-Spielesitzungen, die Sie mit dem Matchmaker verwenden können.

Console

1. Wählen Sie in der GameLiftAmazon-Konsole im Navigationsbereich Matchmaking-Konfigurationen aus.
2. Wechseln Sie zu der AWS Region, in der Sie Ihren Matchmaker erstellen möchten.
3. Wählen Sie auf der Seite Matchmaking-Konfigurationen die Option Matchmaking-Konfiguration erstellen aus.
4. Gehen Sie auf der Seite „Konfigurationsdetails definieren“ unter Matchmaking-Konfigurationsdetails wie folgt vor:

    a. Geben Sie unter Name einen Matchmaker-Namen ein, anhand dessen Sie ihn in einer Liste und in Metriken identifizieren können. Der Matchmaker-Name muss innerhalb der

Region eindeutig sein. Bei Matchmaking-Anfragen wird anhand des Namens und der Region angegeben, welcher Matchmaker verwendet werden soll.

b. (Optional) Fügen Sie unter Beschreibung eine Beschreibung hinzu, um den Matchmaker zu identifizieren.

c. Wählen Sie unter Regelsatz einen Regelsatz aus der Liste aus, den Sie mit dem Matchmaker verwenden möchten. Die Liste enthält alle Regelsätze, die Sie in der aktuellen Region erstellt haben.

d. Wählen Sie als FlexMatchModus Managed for Amazon GameLift Managed Hosting aus. In diesem Modus werden erfolgreiche Spiele FlexMatch an die angegebene Warteschlange für die Spielsitzung weitergeleitet.

e. Wähle AWSunter Region die Region aus, in der du die Warteschlange für Spielsitzungen konfiguriert hast, die du mit dem Matchmaker verwenden möchtest.

f. Wähle unter Warteschlange die Warteschlange für Spielsitzungen aus, die du mit dem Matchmaker verwenden möchtest.

5. Wählen Sie Weiter aus.

6. Gehen Sie auf der Seite Einstellungen konfigurieren unter Matchmaking-Einstellungen wie folgt vor:

a. Lege für das Anforderungs-Timeout die maximale Zeit in Sekunden fest, nach der der Matchmaker für jede Anfrage ein Match abschließen muss. FlexMatchstorniert Matchmaking-Anfragen, die diese Zeit überschreiten.

b. Wählen Sie für den Backfill-Modus einen Modus für die Verarbeitung von Match-Füllungen aus.

- Um die automatische Füllfunktion zu aktivieren, wählen Sie Automatisch.
- Um Ihr eigenes Backfill-Anforderungsmanagement zu erstellen oder die Backfill-Funktion nicht zu verwenden, wählen Sie Manuell.

c. (Optional) Lege für die Anzahl zusätzlicher Spieler die Anzahl der Spielerplätze fest, die in einem Spiel offen bleiben sollen. FlexMatchkann diese Slots in Zukunft mit Spielern füllen.

d. (Optional) Wählen Sie unter Optionen für die Annahme von Spielen unter Annahme erforderlich die Option Erforderlich aus, wenn Sie von jedem Spieler eines vorgeschlagenen Spiels verlangen möchten, dass er aktiv an dem Spiel teilnimmt. Wenn du diese Option auswählst, legst du unter Annahme-Timeout fest, wie lange (in

Sekunden) der Matchmaker auf Zusagen von Spielern warten soll, bevor er das Spiel absagt.

7. (Optional) Gehen Sie unter Einstellungen für die Ereignisbenachrichtigung wie folgt vor:

   a. (Optional) Wählen Sie als SNS-Thema ein Amazon Simple Notification Service (Amazon SNS) -Thema aus, um Benachrichtigungen über Matchmaking-Ereignisse zu erhalten. Wenn du noch kein SNS-Thema eingerichtet hast, kannst du dieses später auswählen, indem du die Matchmaking-Konfiguration bearbeitest. Weitere Informationen finden Sie unter FlexMatchEventbenachrichtigungen einrichten.

   b. (Optional) Geben Sie unter Benutzerdefinierte Eventdaten alle benutzerdefinierten Daten ein, die Sie mit diesem Matchmaker verknüpfen möchten, in der Event-Nachrichtenübermittlung. FlexMatchschließt diese Daten in jedes Ereignis ein, das mit dem Matchmaker verbunden ist.

8. (Optional) Erweitern Sie Zusätzliche Spieldaten und gehen Sie dann wie folgt vor:

   a. (Optional) Geben Sie für Spielsitzungsdaten alle zusätzlichen spielbezogenen Informationen ein, die Sie für neue Spielsitzungen bereitstellen FlexMatch möchten, die mit Spielen beginnen, die mit dieser Matchmaking-Konfiguration erstellt wurden.

   b. (Optional) Fügen Sie für Spieleigenschaften Eigenschaften für Schlüssel-Wert-Paare hinzu, die Informationen über eine neue Spielsitzung enthalten.

9. (Optional) Fügen Sie unter Tags Tags hinzu, mit denen Sie Ihre AWS Ressourcen verwalten und verfolgen können.

10. Wählen Sie Weiter aus.

11. Überprüfen Sie auf der Seite Überprüfen und erstellen Ihre Auswahl und wählen Sie dann Erstellen aus. Nach erfolgreicher Erstellung ist der Matchmaker bereit, Matchmaking-Anfragen anzunehmen.

AWS CLI

Um eine Matchmaking-Konfiguration mit der AWS CLI zu erstellen, öffnen Sie ein Befehlszeilenfenster und verwenden den Befehl create-matchmaking-configuration, um einen neuen Matchmaker zu definieren.

Mit diesem Beispielbefehl wird eine neue Matchmaking-Konfiguration erstellt, die die Zustimmung des Spielers erfordert und das automatische Auffüllen ermöglicht. Es reserviert auch zwei

Spielerplätze, FlexMatch um später Spieler hinzuzufügen, und es stellt einige Spielsitzungsdaten bereit.

```
aws gamelift create-matchmaking-configuration \
    --name "SampleMatchamker123" \
    --description "The sample test matchmaker with acceptance" \
    --flex-match-mode WITH_QUEUE \
    --game-session-queue-arns "arn:aws:gamelift:us-
west-2:111122223333:gamesessionqueue/MyGameSessionQueue" \
    --rule-set-name "MyRuleSet" \
    --request-timeout-seconds 120 \
    --acceptance-required \
    --acceptance-timeout-seconds 30 \
    --backfill-mode AUTOMATIC \
    --notification-target "arn:aws:sns:us-
west-2:111122223333:My_Matchmaking_SNS_Topic" \
    --additional-player-count 2 \
    --game-session-data "key=map,value=winter444"
```

Wenn die Anfrage zur Erstellung der Matchmaking-Konfiguration erfolgreich ist, GameLift gibt Amazon ein MatchmakingConfigurationObjekt mit den Einstellungen zurück, die Sie für den Matchmaker angefordert haben. Der neue Matchmaker ist bereit, Matchmaking-Anfragen anzunehmen.

Erstellen Sie einen Matchmaker für Standalone FlexMatch

Bevor Sie eine Matchmaking-Konfiguration erstellen, erstellen Sie einen Regelsatz, der mit dem Matchmaker verwendet werden soll.

Console

1. Öffnen Sie die GameLift Amazon-Konsole unter https://console.aws.amazon.com/gamelift/home.
2. Wechseln Sie zu der AWS Region, in der Sie Ihren Matchmaker erstellen möchten. Eine Liste der Regionen, die FlexMatch Matchmaking-Konfigurationen unterstützen, finden Sie unterWählen Sie einen Speicherort für den Matchmaker.
3. Wählen Sie FlexMatchim Navigationsbereich Matchmaking-Konfigurationen aus.
4. Wählen Sie auf der Seite Matchmaking-Konfigurationen die Option Matchmaking-Konfiguration erstellen aus.

5. Gehen Sie auf der Seite „Konfigurationsdetails definieren“ unter Matchmaking-Konfigurationsdetails wie folgt vor:

   a. Geben Sie unter Name einen Matchmaker-Namen ein, anhand dessen Sie ihn in einer Liste und in Metriken identifizieren können. Der Matchmaker-Name muss innerhalb der Region eindeutig sein. Bei Matchmaking-Anfragen wird anhand des Namens und der Region angegeben, welcher Matchmaker verwendet werden soll.

   b. (Optional) Fügen Sie unter Beschreibung eine Beschreibung hinzu, um den Matchmaker zu identifizieren.

   c. Wählen Sie unter Regelsatz einen Regelsatz aus der Liste aus, den Sie mit dem Matchmaker verwenden möchten. Die Liste enthält alle Regelsätze, die Sie in der aktuellen Region erstellt haben.

   d. Wählen Sie als FlexMatchModus Standalone. Dies bedeutet, dass Sie über einen benutzerdefinierten Mechanismus verfügen, um neue Spielsitzungen auf einer Hosting-Lösung außerhalb von Amazon zu startenGameLift.

6. Wählen Sie Weiter aus.

7. Gehen Sie auf der Seite Einstellungen konfigurieren unter Matchmaking-Einstellungen wie folgt vor:

   a. Lege für das Anforderungs-Timeout die maximale Zeit in Sekunden fest, nach der der Matchmaker für jede Anfrage ein Match abschließen muss. Matchmaking-Anfragen, die diese Zeit überschreiten, werden abgelehnt.

   b. (Optional) Wählen Sie unter Optionen für die Annahme von Spielen unter Annahme erforderlich die Option Erforderlich aus, wenn Sie von jedem Spieler eines vorgeschlagenen Spiels verlangen möchten, dass er aktiv an dem Spiel teilnimmt. Wenn du diese Option auswählst, legst du unter Annahme-Timeout fest, wie lange (in Sekunden) der Matchmaker auf Zusagen von Spielern warten soll, bevor er das Spiel absagt.

8. (Optional) Gehen Sie unter Einstellungen für die Ereignisbenachrichtigung wie folgt vor:

   a. (Optional) Wählen Sie für das SNS-Thema ein Amazon SNS-Thema aus, um Benachrichtigungen über Matchmaking-Ereignisse zu erhalten. Wenn du noch kein SNS-Thema eingerichtet hast, kannst du dieses später auswählen, indem du die Matchmaking-Konfiguration bearbeitest. Weitere Informationen finden Sie unter FlexMatchEventbenachrichtigungen einrichten.

b. (Optional) Geben Sie unter Benutzerdefinierte Eventdaten alle benutzerdefinierten Daten ein, die Sie mit diesem Matchmaker verknüpfen möchten, in der Event-Nachrichtenübermittlung. FlexMatchschließt diese Daten in jedes Ereignis ein, das mit dem Matchmaker verbunden ist.

9. (Optional) Fügen Sie unter Tags Tags hinzu, mit denen Sie Ihre AWS Ressourcen verwalten und verfolgen können.
10. Wählen Sie Weiter aus.
11. Überprüfen Sie auf der Seite Überprüfen und erstellen Ihre Auswahl und wählen Sie dann Erstellen aus. Nach erfolgreicher Erstellung ist der Matchmaker bereit, Matchmaking-Anfragen anzunehmen.

AWS CLI

Um eine Matchmaking-Konfiguration mit der AWS CLI zu erstellen, öffnen Sie ein Befehlszeilenfenster und verwenden den Befehl create-matchmaking-configuration, um einen neuen Matchmaker zu definieren.

Mit diesem Beispielbefehl wird eine neue Matchmaking-Konfiguration für einen eigenständigen Matchmaker erstellt, für die die Zustimmung des Spielers erforderlich ist.

```
aws gamelift create-matchmaking-configuration \
    --name "SampleMatchamker123" \
    --description "The sample test matchmaker with acceptance" \
    --flex-match-mode STANDALONE \
    --rule-set-name "MyRuleSetOne" \
    --request-timeout-seconds 120 \
    --acceptance-required \
    --acceptance-timeout-seconds 30 \
    --notification-target "arn:aws:sns:us-
west-2:111122223333:My_Matchmaking_SNS_Topic"
```

Wenn die Anfrage zur Erstellung der Matchmaking-Konfiguration erfolgreich ist, GameLift gibt Amazon ein MatchmakingConfigurationObjekt mit den Einstellungen zurück, die Sie für den Matchmaker angefordert haben. Der neue Matchmaker ist bereit, Matchmaking-Anfragen anzunehmen.

Bearbeiten Sie eine Matchmaking-Konfiguration

Um eine Matchmaking-Konfiguration zu bearbeiten, wählen Sie Matchmaking-Konfigurationen in der Navigationsleiste und wählen Sie die Konfiguration aus, die Sie bearbeiten möchten. Sie können jedes Feld in einer vorhandenen Konfiguration mit Ausnahme seines Namens aktualisieren.

Bei der Aktualisierung eines Konfigurationsregelsatzes kann ein neuer Regelsatz aus den folgenden Gründen inkompatibel sein, wenn aktive Matchmaking-Tickets vorhanden sind:

- Neue oder andere Teamnamen oder Anzahl der Teams
- Neue Spielerattribute
- Änderungen an bestehenden Spieler-Attributtypen

Um eine dieser Änderungen an Ihrem Regelsatz vorzunehmen, erstellen Sie eine neue Matchmaking-Konfiguration mit dem aktualisierten Regelsatz.

FlexMatchEventbenachrichtigungen einrichten

Mithilfe von Eventbenachrichtigungen können Sie den Status einzelner Matchmaking-Anfragen verfolgen. Alle Spiele, die sich in der Produktion befinden oder sich in der Vorproduktion befinden und bei denen viele Matchmaking-Aktivitäten stattfinden, sollten Event-Benachrichtigungen verwenden.

Es gibt zwei Möglichkeiten, Ereignisbenachrichtigungen einzurichten.

- Bitten Sie Ihren Matchmaker, Event-Benachrichtigungen zu einem Amazon Simple Notification Service (Amazon SNS) -Thema zu veröffentlichen.
- Verwenden Sie automatisch veröffentlichte EventBridge Amazon-Events und die zugehörigen Tools zur Verwaltung von Veranstaltungen.

Eine Liste der FlexMatch Ereignisse, die Amazon ausgibtGameLift, finden Sie unterFlexMatchMatchmaking-Veranstaltungen.

EventBridgeEreignisse einrichten

Amazon veröffentlicht GameLift automatisch alle Matchmaking-Events auf AmazonEventBridge. Mit können Sie Regeln einrichtenEventBridge, nach denen Matchmaking-Ereignisse zur Verarbeitung an Ziele weitergeleitet werden. Sie können beispielsweise eine Regel festlegen, um das Ereignis

"PotentialMatchCreated" an eine AWS Lambda Funktion weiterzuleiten, die die Annahme von Spielern verwaltet. Weitere Informationen finden Sie unter Was ist AmazonEventBridge?

> **Note**
>
> Wenn Sie Ihre Matchmaker konfigurieren, lassen Sie das Feld für das Benachrichtigungsziel leer oder verweisen Sie auf ein SNS-Thema, wenn Sie EventBridge sowohl Amazon SNS als auch Amazon SNS verwenden möchten.

Ein Amazon SNS-Thema einrichten

Sie können Amazon veranlassen, alle Ereignisse, die ein FlexMatch Matchmaker generiert, zu einem Amazon SNS-Thema zu GameLift veröffentlichen.

So erstellen Sie ein SNS-Thema für GameLift Amazon-Eventbenachrichtigungen

1. Öffnen Sie die Amazon-SNS-Konsole.
2. Wählen Sie im Navigationsbereich Topics (Themen) aus.
3. Klicken Sie auf der Seite Themen auf Thema erstellen.
4. Erstellen Sie ein Thema in der -Konsole. Weitere Informationen finden Sie unter So erstellen Sie ein Thema mithilfe des AWS Management Console im Amazon Simple Notification Service Developer Guide.
5. Wählen Sie auf der Detailseite für Ihr Thema die Option Bearbeiten aus.
6. (Optional) Erweitern Sie auf der Seite Bearbeiten für Ihr Thema die Access-Richtlinie und fügen Sie dann die fett formatierte Syntax aus der folgenden AWS Identity and Access Management (IAM-) Richtlinienanweisung am Ende Ihrer vorhandenen Richtlinie hinzu. (Die gesamte Richtlinie wird hier der Übersichtlichkeit halber dargestellt.) Verwenden Sie unbedingt die Angaben zum Amazon-Ressourcennamen (ARN) für Ihr eigenes SNS-Thema und die GameLift Amazon-Matchmaking-Konfiguration.

```
{
  "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:matchmakingconfiguration/
your_configuration_name"
        }
      }
    }
  ]
}
```

7. Wählen Sie Änderungen speichern aus.

Richten Sie ein SNS-Thema mit serverseitiger Verschlüsselung ein

Sie können serverseitige Verschlüsselung (SSE) verwenden, um vertrauliche Daten in verschlüsselten Themen zu speichern. SSE schützt den Inhalt von Nachrichten in Amazon SNS-Themen mit in AWS Key Management Service (AWS KMS) verwalteten Schlüsseln. Weitere Informationen zur serverseitigen Verschlüsselung mit Amazon SNS finden Sie unter Encryption at Rest im Amazon Simple Notification Service Developer Guide.

Lesen Sie die folgenden Themen, um ein SNS-Thema mit serverseitiger Verschlüsselung einzurichten:

- Schlüssel im AWS Key Management ServiceEntwicklerhandbuch erstellen
- SSE für ein Thema im Amazon Simple Notification Service Developer Guide aktivieren

Verwenden Sie beim Erstellen Ihres KMS-Schlüssels die folgende KMS-Schlüsselrichtlinie:

```
{
  "Effect": "Allow",
  "Principal": {
    "Service": "gamelift.amazonaws.com"
  },
  "Action": [
      "kms:Decrypt",
      "kms:GenerateDataKey"
  ],
  "Resource": "*",
  "Condition": {
      "ArnLike": {
        "aws:SourceArn":
 "arn:aws:gamelift:your_region:your_account:matchmakingconfiguration/
your_configuration_name"
      },
      "StringEquals": {
        "kms:EncryptionContext:aws:sns:topicArn":
 "arn:aws:sns:your_region:your_account:your_sns_topic_name"
      }
  }
}
```

Konfigurieren Sie ein Themenabonnement, um eine Lambda-Funktion aufzurufen

Sie können eine Lambda-Funktion mithilfe von Ereignisbenachrichtigungen aufrufen, die in Ihrem Amazon SNS-Thema veröffentlicht wurden. Achten Sie bei der Konfiguration des Matchmakers darauf, als Benachrichtigungsziel den ARN Ihres SNS-Themas festzulegen.

Die folgende AWS CloudFormation Vorlage konfiguriert ein Abonnement für ein SNS-Thema mit dem Namen, um eine `MyFlexMatchEventTopic` Lambda-Funktion mit dem Namen aufzurufen. `FlexMatchEventHandlerLambdaFunction` Die Vorlage erstellt eine IAM-Berechtigungsrichtlinie, die es Amazon ermöglicht, GameLift zum SNS-Thema zu schreiben. Die Vorlage fügt dann Berechtigungen für das SNS-Thema hinzu, um die Lambda-Funktion aufzurufen.

```
FlexMatchEventTopic:
  Type: "AWS::SNS::Topic"
  Properties:
    KmsMasterKeyId: alias/aws/sns #Enables server-side encryption on the topic using an
 AWS managed key
    Subscription:
      - Endpoint: !GetAtt FlexMatchEventHandlerLambdaFunction.Arn
        Protocol: lambda
    TopicName: MyFlexMatchEventTopic

FlexMatchEventTopicPolicy:
  Type: "AWS::SNS::TopicPolicy"
  DependsOn: FlexMatchEventTopic
  Properties:
    PolicyDocument:
      Version: "2012-10-17"
      Statement:
        - Effect: Allow
          Principal:
            Service: gamelift.amazonaws.com
          Action:
            - "sns:Publish"
          Resource: !Ref FlexMatchEventTopic
    Topics:
      - Ref: FlexMatchEventTopic

FlexMatchEventHandlerLambdaPermission:
  Type: "AWS::Lambda::Permission"
  Properties:
```

```
    Action: "lambda:InvokeFunction"
    FunctionName: !Ref FlexMatchEventHandlerLambdaFunction
    Principal: sns.amazonaws.com
    SourceArn: !Ref FlexMatchEventTopic
```

Spiele vorbereiten für FlexMatch

Verwenden Sie Amazon GameLiftFlexMatch, um Ihren Spielen Spieler-Matchmaking-Funktionen hinzuzufügen. FlexMatchist mit den verwalteten GameLift Amazon-Lösungen für benutzerdefinierte Spieleserver und Echtzeitserver verfügbar.

FlexMatch verbindet den Matchmaking-Service mit einer anpassbaren Regel-Engine. Auf diese Weise können Sie festlegen, wie Sie Spieler basierend auf Spielerattributen und Spielmodi zusammenstellen, die für Ihr Spiel sinnvoll sind, und es FlexMatch überlassen, Spielergruppen zu bilden und sie in Spielen zu platzieren. Weitere Details zum benutzerdefinierten Matchmaking finden Sie unter FlexMatch Beispiele für Regelsätze.

FlexMatch basiert auf der Warteschlangenfunktion. Sobald ein Match erstellt wurde, übergibt FlexMatch die Details einer Warteschlange Ihrer Wahl. Die Warteschlange sucht nach verfügbaren Hosting-Ressourcen auf Ihren GameLift Amazon-Flotten und startet eine neue Spielsitzung für das Spiel.

Die Themen in diesem Abschnitt gehen darauf ein, wie Sie Ihre Spieleserver und Spiele-Clients um Matchmaking-Unterstützung erweitern. Informationen zum Erstellen eines Matchmaker für Ihr Spiel finden Sie unter Aufbau eines Amazon-Matchmakers GameLift FlexMatch. Weitere Informationen zur Funktionsweise von FlexMatch finden Sie unter So GameLift FlexMatch funktioniert Amazon.

FlexMatchZu einem Spielclient hinzufügen

In diesem Thema wird beschrieben, wie Sie Ihren clientseitigen Spieledienssten FlexMatch Matchmaking-Unterstützung hinzufügen können. Der Prozess ist im Wesentlichen derselbe, unabhängig davon, ob Sie es FlexMatch mit Amazon GameLift Managed Hosting oder mit einer anderen Hosting-Lösung verwenden. Weitere Informationen zu FlexMatch und zum Einrichten eines benutzerdefinierten Matchmakers für Ihre Spiele finden Sie unter folgenden Themen:

- FlexMatchIntegration mit GameLift Amazon-Hosting
- So GameLift FlexMatch funktioniert Amazon
- Aufbau eines Amazon-Matchmakers GameLift FlexMatch
- FlexMatch Beispiele für Regelsätze

Füge die folgende Funktionalität hinzu, um FlexMatch Matchmaking in deinem Spiel zu aktivieren:

- Bereite dich darauf vor, Matchmaking für einen oder mehrere Spieler anzufordern (erforderlich).
- Verfolge den Status von Matchmaking-Anfragen (erforderlich).
- Bitten Sie den Spieler um die Annahme eines vorgeschlagenen Spiels (optional).
- Nachdem eine Spielsitzung für das neue Spiel erstellt wurde, rufen Sie die Verbindungsinformationen des Spielers ab und treten Sie dem Spiel bei.

Bereite dich darauf vor, Matchmaking für Spieler anzufordern

Wir empfehlen dringend, dass dein Spielclient Matchmaking-Anfragen über einen clientseitigen Spieledienst stellt. Durch Verwenden einer vertrauenswürdigen Quelle können Sie sich leichter gegen Hacking-Versuche und gefälschte Spielerdaten schützen. Wenn Ihr Spiel einen Sitzungsverzeichnisdienst hat, ist dies eine gute Option für die Bearbeitung von Matchmaking-Anforderungen.

Führen Sie zur Vorbereitung Ihres Client-Service die folgenden Aufgaben aus:

- Fügen Sie die GameLift Amazon-API hinzu. Ihr Kundenservice verwendet Funktionen der GameLift Amazon-API, die Teil des AWS SDK ist. Weitere Informationen zum GameLift SDK und zum Herunterladen der neuesten Version finden Sie unter Amazon AWS SDKs for Client Services. Fügen Sie dieses SDK zum Service-Projekt Ihres Spiele-Clients hinzu.
- Richten Sie ein Matchmaking-Ticketsystem ein. Allen Matchmaking-Anforderungen muss eine eindeutige Ticket-ID zugewiesen werden. Sie benötigen einen Methode, mit der Sie eindeutige IDs generieren und sie neuen Match-Anforderungen zuweisen können. Für eine Ticket-ID kann ein beliebiges Zeichenfolgenformat mit maximal 128 Zeichen verwendet werden.
- Fordern Sie Matchmaker-Informationen an. Fordern Sie den Namen der Matchmaking-Konfiguration an, die Sie verwenden möchten. Außerdem benötigen Sie die Liste der erforderlichen Spielerattribute des Matchmakers, die in der Regelmenge des Matchmakers definiert sind.
- Rufen Sie Spielerdaten ab. Richten Sie eine Möglichkeit ein, relevante Daten für jeden Spieler abzurufen. Dazu gehören Spieler-ID, Spielerattributwerte und aktualisierte Latenzdaten für jede Region, in der der Spieler wahrscheinlich in einem Spiel platziert wird.
- (Optional) Aktivieren Sie Match-Backfill. Entscheiden Sie, wie Sie Ihre vorhandenen Match-Spiele auffüllen möchten. Wenn als Backfill-Modus Ihrer Matchmaker „manuell“ eingestellt ist, ist es ratsam, Backfill-Unterstützung zu Ihrem Spiel hinzuzufügen. Wenn als Backfill-Modus „automatisch“ eingestellt ist, benötigen Sie möglicherweise eine Methode, mit der Sie ihn für einzelne Spielsitzungen ausschalten können. Weitere Informationen zum Verwalten von Match-Backfill finden Sie in Füllt bestehende Spiele auf mit FlexMatch.

Matchmaking für Spieler anfragen

Fügen Sie Code zum Erstellen und Verwalten von Matchmaking-Anforderungen an einen FlexMatch-Matchmaker zu Ihrem Client-Service hinzu. Das Anfordern von FlexMatch Matchmaking ist für Spiele, die FlexMatch mit GameLiftmanaged Amazon-Hosting verwendet werden, und für Spiele, die FlexMatch als eigenständige Lösung verwendet werden, identisch.

Erstellen einer Matchmaking-Anforderung:

- Rufen Sie die GameLift Amazon-API auf StartMatchmaking. Jede Anforderung muss die folgenden Informationen enthalten.

  Matchmaker

  Der Name der Matchmaking-Konfiguration, die für die Anfrage verwendet werden soll. FlexMatchplatziert jede Anfrage in den Pool für den angegebenen Matchmaker, und die Anfrage wird basierend auf der Konfiguration des Matchmakers bearbeitet. Dies umfasst das Erzwingen eines Zeitlimits, ob die Spieler-Akzeptanz von Matches angefordert werden soll, welche Warteschlange beim Platzieren einer resultierenden Spielsitzung verwendet werden soll usw. Weitere Informationen zu Matchmakern und Regelsätzen finden Sie unter Entwerfen eines FlexMatch Matchmakers.

  Ticket-ID

  Eine der Anforderung zugewiesene eindeutige Ticket-ID. Alles im Zusammenhang mit der Anforderung, u. a. auch Ereignisse und Benachrichtigungen, referenziert die Ticket-ID.

  Spielerdaten

  Liste der Spieler, für die Sie ein Match erstellen möchten. Wenn einer der Spieler in der Anforderung die Match-Anforderungen nicht erfüllt, führt die Matchmaking-Anforderung basierend auf den Match-Regeln und Latenzminimalen nie zu einem erfolgreichen Match. Sie können bis zu zehn Spieler in eine Match-Anforderung aufnehmen. Wenn mehrere Spieler in einer Anforderung vorhanden sind, versucht FlexMatch, ein einzelnes Match zu erstellen und alle Spieler demselben Team zuzuweisen (zufällig ausgewählt). Wenn eine Anforderung zu viele Spieler enthält, um in eines der Match-Teams zu passen, wird die Anforderung nicht abgeglichen. Wenn Sie beispielsweise Ihren Matchmaker so eingerichtet haben, dass 2v2-Matches (zwei Teams mit zwei Spielern) erstellt werden, können Sie keine Matchmaking-Anforderung senden, die mehr als zwei Spieler enthält.

> **Note**
>
> Ein Spieler (identifiziert durch die Spieler-ID) kann jeweils nur in eine aktive Matchmaking-Anforderung aufgenommen werden. Wenn Sie eine neue Anforderung für einen Spieler erstellen, werden alle aktiven Matchmaking-Tickets mit derselben Spieler-ID automatisch storniert.

Schließen Sie für jeden aufgelisteten Spieler die folgenden Daten ein:

- Spieler-ID — Jeder Spieler muss eine eindeutige Spieler-ID haben, die du generierst. Siehe Generieren von Spieler-IDs.
- Spielerattribute — Wenn der verwendete Matchmaker Spielerattribute verlangt, muss die Anfrage diese Attribute für jeden Spieler angeben. Die erforderlichen Spielerattribute sind im Regelsatz des Matchmakers definiert, in dem auch der Datentyp für das Attribut angegeben wird. Ein Spielerattribut ist nur dann optional, wenn der Regelsatz einen Standardwert für das Attribut angibt. Wenn die Match-Anforderung nicht die erforderlichen Attribute für alle Spieler bereitstellt, kann die Matching-Anforderung niemals erfolgreich sein. Weitere Informationen zu Matchmaker-Regelsätzen und Spielerattributen finden Sie unter Erstellen Sie einen FlexMatch Regelsatz und FlexMatch Beispiele für Regelsätze.
- Spielerlatenzen — Wenn der verwendete Matchmaker eine Regel für die Spielerlatenz hat, muss die Anfrage die Latenz für jeden Spieler angeben. Bei Player-Latenzdaten handelt es sich um eine Liste mit einem oder mehreren Werten pro Spieler. Dadurch wird die Latenzerfahrung des Spielers für Regionen in der Warteschlange des Matchmakers darstellt. Wenn in der Anforderung keine Latenzwerte für einen Spieler enthalten sind, kann der Spieler in kein Match aufgenommen werden. Die Anforderung schlägt fehl.

Abrufen von Match-Anforderungsdetails:

- Sobald eine Spielanfrage gesendet wurde, können Sie die Details der Anfrage einsehen, indem Sie DescribeMatchmakingmit der Ticket-ID der Anfrage anrufen. Dieser Aufruf gibt Informationen zur Anforderung zurück, u. a. den aktuellen Status. Sobald eine Anforderung erfolgreich abgeschlossen wurde, enthält das Ticket die Informationen, die der Spiele-Client zum Verbinden mit dem Match benötigt.

Abbrechen einer Match-Anforderung:

- Sie können eine Matchmaking-Anfrage jederzeit stornieren, indem Sie StopMatchmakingmit der Ticket-ID der Anfrage anrufen.

Verfolgen Sie Matchmaking-Events

Richten Sie Benachrichtigungen ein, um Ereignisse zu verfolgen, die Amazon für Matchmaking-Prozesse GameLift ausgibt. Sie können Benachrichtigungen entweder direkt einrichten, indem Sie ein SNS-Thema erstellen, oder indem Sie Amazon EventBridge verwenden. Weitere Informationen zur Einrichtung von Benachrichtigungen finden Sie unter FlexMatchEventbenachrichtigungen einrichten. Nachdem Sie Benachrichtigungen eingerichtet haben, fügen Sie Ihrem Client-Service einen Listener hinzufügen, der die Ereignisse erkennt und gegebenenfalls auf sie reagiert.

Es ist auch eine gute Idee, Benachrichtigungen zu sichern, indem Sie nach einem erheblichen Zeitraum ohne Benachrichtigung regelmäßig Statusaktualisierungen abfragen. Um eine Beeinträchtigung der Matchmaking-Leistung zu minimieren, stellen Sie sicher, mit dem Abfragen mindestens 30 Sekunden ab dem Senden des Matchmaking-Tickets oder der zuletzt erhaltenen Benachrichtigung zu warten.

Rufen Sie ein Matchmaking-Anfrageticket, einschließlich des aktuellen Status, an, indem Sie DescribeMatchmakingmit der Ticket-ID der Anfrage anrufen. Wir empfehlen Abfragen nicht öfter als einmal alle 10 Sekunden. Diese Methode ist nur für Entwicklungsszenarien mit geringem Datenaufkommen bestimmt.

> **Note**
>
> Vor einer Matchmaking-Nutzung mit hohem Datenaufkommen, wie z. B. bei Vorproduktions-Lasttests, sollten Sie Ihr Spiel mit Ereignisbenachrichtigungen einrichten. Bei allen veröffentlichen Versionen von Spielen sollten unabhängig vom Datenaufkommen Benachrichtigungen verwendet werden. Die Methode der stetigen Abfragen eignet sich nur für Spiele in Entwicklung mit geringer Matchmaking-Nutzung.

Spielerzusage anfragen

Wenn Sie einen Matchmaker mit aktivierter Spieler-Akzeptanz verwenden, fügen Sie Ihrem Client-Service Code zur Verwaltung des Spieler-Akzeptanzvorgangs hinzu. Der Prozess der Verwaltung

der Spielerakzeptanz ist bei Spielen, die FlexMatch mit von Amazon GameLift verwaltetem Hosting verwendet werden, und bei Spielen, die FlexMatch als eigenständige Lösung verwendet werden, identisch.

Anfordern der Spieler-Akzeptanz für ein vorgeschlagenes Match:

1. Stellen Sie fest, wenn ein vorgeschlagenes Match Spieler-Akzeptanz benötigt. Überwachen Sie das Matchmaking-Ticket, um zu erkennen, wann sich der Status in `REQUIRES_ACCEPTANCE` ändert. Eine Änderung dieses Status löst das FlexMatch Ereignis aus`MatchmakingRequiresAcceptance`.
2. Fordern Sie Akzeptanz von allen Spielern an. Erstellen Sie eine Methode, mit der Sie jedem Spieler im Matchmaking-Ticket die vorgeschlagenen Match-Details präsentieren können. Spieler müssen angeben können, ob sie das vorgeschlagene Match annehmen oder ablehnen. Sie können die Spieldetails abrufen, indem Sie anrufen DescribeMatchmaking. Spieler haben begrenzt Zeit zu reagieren, bevor der Matchmaker den Match-Vorschlag zurückzieht und weitersucht.
3. Melden Sie die Spielerantworten an FlexMatch. Meldet die Antworten der Spieler, indem ihr entweder AcceptMatchmit „Annehmen“ oder „Ablehnen“ anruft. Alle Spieler in einer Matchmaking-Anforderung müssen das Match akzeptieren, damit es weitergeführt wird.
4. Verarbeiten Sie Tickets mit fehlgeschlagenen Zusagen. Eine Anforderung schlägt fehl, wenn ein oder mehrere Spieler im vorgeschlagenen Match entweder das Match abgelehnt oder nicht innerhalb des Zeitlimits für die Akzeptanz reagiert haben. Tickets für Spieler, die das Spiel angenommen haben, werden automatisch in den Ticketpool zurückgeführt. Tickets für Spieler, die das Spiel nicht akzeptiert haben, erhalten den Status FAILURE und werden nicht mehr bearbeitet. Bei Tickets mit mehreren Spielern gilt: Wenn ein Spieler auf dem Ticket das Spiel nicht akzeptiert hat, ist das gesamte Ticket ungültig.

Stellen Sie eine Verbindung zu einem Spiel her

Füge deinem Kundenservice Code hinzu, um ein erfolgreich gefundenes Match (Status `COMPLETED` oder Ereignis`MatchmakingSucceeded`) zu bearbeiten. Dies umfasst die Benachrichtigung der Spieler des Matches und die Übergabe von Verbindungsinformationen an deren Spiele-Clients.

Bei Spielen, die von Amazon GameLift verwaltetes Hosting verwenden, werden die Verbindungsinformationen zur Spielsitzung dem Matchmaking-Ticket hinzugefügt, wenn eine Matchmaking-Anfrage erfolgreich erfüllt wurde. Rufen Sie ein ausgefülltes Matchmaking-Ticket an DescribeMatchmaking. Verbindungsinformationen umfassen die IP-Adresse und den Port der

Spielsitzung sowie eine Spielersitzungs-ID für jede Spieler-ID. Weitere Informationen finden Sie unter GameSessionConnectionInfo. Ihr Spielclient kann diese Informationen verwenden, um sich direkt mit der Spielsitzung für das Spiel zu verbinden. Die Verbindungsanfrage sollte eine Spielersitzungs-ID und eine Spieler-ID enthalten. Diese Daten verknüpfen den verbundenen Spieler mit den Spieldaten der Spielsitzung, zu denen auch Teamzuweisungen gehören (siehe GameSession).

Für Spiele, die andere Hosting-Lösungen verwenden, einschließlich Amazon GameLift FleetIQ, müssen Sie einen Mechanismus einbauen, der es Match-Spielern ermöglicht, sich mit der entsprechenden Spielsitzung zu verbinden.

Beispiel für Matchmaking-Anfragen

Die folgenden Codefragmente erstellen Matchmaking-Anfragen für verschiedene Matchmaker. Wie beschrieben, muss eine Anforderung die Spielerattribute bereitstellen, die laut dem definierten Matchmaker-Regelsatz für den aktuell verwendeten Matchmaker erforderlich sind. Das bereitgestellte Attribut muss den gleichen Datentyp, die gleiche Zahl (N) oder die gleiche Zeichenfolge (S) wie im Regelsatz definiert verwenden.

```
# Uses matchmaker for two-team game mode based on player skill level
def start_matchmaking_for_cowboys_vs_aliens(config_name, ticket_id, player_id, skill,
 team):
    response = gamelift.start_matchmaking(
        ConfigurationName=config_name,
        Players=[{
            "PlayerAttributes": {
                "skill": {"N": skill}
            },
            "PlayerId": player_id,
            "Team": team
        }],
        TicketId=ticket_id)

# Uses matchmaker for monster hunter game mode based on player skill level
def start_matchmaking_for_players_vs_monster(config_name, ticket_id, player_id, skill,
 is_monster):
    response = gamelift.start_matchmaking(
        ConfigurationName=config_name,
        Players=[{
            "PlayerAttributes": {
                "skill": {"N": skill},
                "desiredSkillOfMonster": {"N": skill},
```

```
                "wantsToBeMonster": {"N": int(is_monster)}
            },
            "PlayerId": player_id
        }],
        TicketId=ticket_id)

# Uses matchmaker for brawler game mode with latency
def start_matchmaking_for_three_team_brawler(config_name, ticket_id, player_id, skill,
 role):
    response = gamelift.start_matchmaking(
        ConfigurationName=config_name,
        Players=[{
            "PlayerAttributes": {
                "skill": {"N": skill},
                "character": {"S": [role]},
            },
            "PlayerId": player_id,
            "LatencyInMs": { "us-west-2": 20}
        }],
        TicketId=ticket_id)

# Uses matchmaker for multiple game modes and maps based on player experience
def start_matchmaking_for_multi_map(config_name, ticket_id, player_id, skill, maps,
 modes):
    response = gamelift.start_matchmaking(
        ConfigurationName=config_name,
        Players=[{
            "PlayerAttributes": {
                "experience": {"N": skill},
                "gameMode": {"SL": modes},
                "mapPreference": {"SL": maps}
            },
            "PlayerId": player_id
        }],
        TicketId=ticket_id)
```

FlexMatchZu einem von Amazon GameLift gehosteten Spieleserver hinzufügen

In diesem Thema wird beschrieben, wie Sie benutzerdefinierten Spieleservern, die Amazon GameLift Managed Hosting verwenden, FlexMatch Matchmaking-Unterstützung hinzufügen. Weitere Informationen zum Hinzufügen von FlexMatch zu Ihren Spielen finden Sie in den folgenden Themen:

- So GameLift FlexMatch funktioniert Amazon
- FlexMatchIntegration mit GameLift Amazon-Hosting

Bei den Informationen in diesem Thema wird davon ausgegangen, dass Sie das Amazon GameLift Server SDK erfolgreich in Ihr Gameserverprojekt integriert haben, wie unter Amazon GameLift zu Ihrem Gameserver hinzufügen beschrieben. Wenn dies abgeschlossen ist, stehen die meisten der benötigten Mechanismen zur Verfügung. Die Abschnitte in diesem Thema behandeln die verbleibenden Aktivitäten für Spiele, die mit FlexMatch eingerichtet wurden.

Richte deinen Gameserver für Matchmaking ein

Um Ihren Spielserver so einzurichten, dass er mit dem Spiel-Matching umgehen kann, führen Sie die folgenden Aufgaben aus.

1. Starten Sie Spielsitzungen, die mit Matchmaking erstellt wurden. Um eine neue Spielsitzung anzufordern, GameLift sendet Amazon eine `onStartGameSession()` Anfrage mit einem Spielesitzungsobjekt an Ihren Spieleserver (siehe GameSession). Ihr Spielserver verwendet die Informationen der Spielsitzung, einschließlich der angepassten Spieldaten, um die angeforderte Spielsitzung zu starten. Weitere Informationen findest du unter Eine Spielsitzung starten.

   Für Match-Spiele enthält das Spielsitzungsobjekt auch eine Reihe von Matchmaker-Daten. Matchmaker-Daten enthalten Informationen, die Ihr Spieleserver benötigt, um eine neue Spielsitzung für das Match zu starten. Dazu gehören die Teamstruktur des Matches, Teamzuweisungen und bestimmte Spielerattribute, die für Ihr Spiel relevant sein könnten. Ihr Spiel könnte beispielsweise basierend auf der durchschnittlichen Spieler-Qualifikationsstufe bestimmte Features oder Levels freischalten. Sie können basierend auf den Präferenzen des Spielers aber auch eine Zuordnung wählen. Weitere Informationen finden Sie unter Arbeiten Sie mit Matchmaker-Daten.
2. Verarbeitung von Spieler-Verbindungen. Beim Herstellen einer Verbindung zu einem passenden Spiel verweist ein Spielclient auf eine Spieler-ID und eine Spielersitzungs-ID (siehe Einen neuen Spieler validieren). Ihr Spieleserver verwendet die Spieler-ID, um einen eingehenden Spieler mit Spielerinformationen in den Matchmaker-Daten zu verknüpfen. Matchmaker-Daten identifizieren die Teamzuordnung eines Spielers und können weitere Informationen für die korrekte Darstellung des Spielers im Spiel liefern.
3. Melden von Spielern, die ein Spiel verlassen. Vergewissere dich, dass dein Gameserver die Server-API aufruft`RemovePlayerSession()`, um ausgeschiedene Spieler zu melden (siehe Ende einer Spielersitzung melden). Dieser Schritt ist wichtig, wenn Sie FlexMatch-Backfill zum

Füllen verfügbarer Spielerplätze in bestehenden Spielen verwenden. Dies ist von entscheidender Bedeutung, wenn Ihr Spiel Backfill-Anforderungen über einen clientseitigen Spieleservice initiiert. Weitere Informationen zur Implementierung von FlexMatch-Backfill finden Sie unter Füllt bestehende Spiele auf mit FlexMatch.

4. Anfordern von neuen Spielern für bestehende Spielsitzungen (optional). Entscheiden Sie, wie Sie Ihre vorhandenen Match-Spiele auffüllen möchten. Wenn als Backfill-Modus Ihrer Matchmaker „manuell“ eingestellt ist, ist es ratsam, Backfill-Unterstützung zu Ihrem Spiel hinzuzufügen. Wenn als Backfill-Modus „automatisch“ eingestellt ist, benötigen Sie möglicherweise eine Methode, mit der Sie ihn für einzelne Spielsitzungen ausschalten können. So möchten Sie beispielsweise vielleicht das Backfilling einer Spielsitzung stoppen, sobald ein bestimmter Punkt im Spiel erreicht wurde. Weitere Informationen zum Verwalten von Match-Backfill finden Sie in Füllt bestehende Spiele auf mit FlexMatch.

Arbeiten Sie mit Matchmaker-Daten

Ihr Gameserver muss in der Lage sein, die Spielinformationen in einem GameSessionObjekt zu erkennen und zu verwenden. Der GameLift Amazon-Dienst leitet diese Objekte an Ihren Spieleserver weiter, wenn eine Spielesitzung gestartet oder aktualisiert wird. Zu den Kerninformationen der Spielsitzung gehören die ID und der Name der Spielsitzung, die maximale Spieleranzahl, Verbindungsinformationen und benutzerdefinierte Spieldaten (falls vorhanden).

Bei Spielsitzungen, die mit FlexMatch erstellt werden, enthält das `GameSession`-Objekt auch eine Reihe von Matchmaker-Daten. Zusätzlich zu einer eindeutigen Match-ID identifiziert es den Matchmaker, mit dem das Match erstellt wurde, und beschreibt die Teams, Teamzuweisungen und Spieler. Es enthält die Spielerattribute aus der ursprünglichen Matchmaking-Anforderung (siehe Player-Objekt). Wenn Sie Latenzdaten von aktuellen Spielern benötigen (z. B. für das Match-Backfill), empfehlen wir Ihnen, neue Daten abzurufen.

> **Note**
>
> Matchmaker-Daten geben den vollständigen Matchmaking-Konfigurations-ARN an, der den Konfigurationsnamen, das AWS Konto und die Region identifiziert. Wenn Match-Backfill von einem Spiele-Client oder -Service angefordert wird, wird nur der Konfigurationsname benötigt. Sie können den Konfigurationsnamen extrahieren, indem Sie die Zeichenfolge nach „:matchmakingconfiguration/“ analysieren. Im gezeigten Beispiel lautet der Name der Matchmaking-Konfiguration "MyMatchmakerConfig“.

Die folgenden JSON-Daten zeigen einen typischen Satz von Matchmaker-Daten. Dieses Beispiel beschreibt ein Spiel mit zwei Spielern, bei dem die Spieler auf der Grundlage von Skill-Bewertungen und der höchsten erreichten Stufe aufeinander abgestimmt sind. Der Matchmaker hat außerdem auf Basis des Charakters nach passenden Spielern gesucht. So wurde sichergestellt, dass die Spieler, die sich in einem Match befinden, mindestens eine bevorzugte Karte gemeinsam haben. In diesem Fall sollte der Spieleserver in der Lage sein, die am meisten bevorzugte Zuordnung zu bestimmen und sie in der Spielsitzung zu verwenden.

```
{
 "matchId":"1111aaaa-22bb-33cc-44dd-5555eeee66ff",
 "matchmakingConfigurationArn":"arn:aws:gamelift:us-
west-2:111122223333:matchmakingconfiguration/MyMatchmakerConfig",
 "teams":[
    {"name":"attacker",
  "players":[
           {"playerId":"4444dddd-55ee-66ff-77aa-8888bbbb99cc",
   "attributes":{
    "skills":{
     "attributeType":"STRING_DOUBLE_MAP",
     "valueAttribute":{"Body":10.0,"Mind":12.0,"Heart":15.0,"Soul":33.0}}
   }
  }]
 },{
  "name":"defender",
  "players":[{
   "playerId":"3333cccc-44dd-55ee-66ff-7777aaaa88bb",
   "attributes":{
    "skills":{
     "attributeType":"STRING_DOUBLE_MAP",
     "valueAttribute":{"Body":11.0,"Mind":12.0,"Heart":11.0,"Soul":40.0}}
   }
  }]
 }]
}
```

Füllt bestehende Spiele auf mit FlexMatch

Match-Backfill nutzt Ihre FlexMatch-Mechanismen, um neue Spieler für vorhandene zugeordnete Spielesitzungen zu finden. Du kannst zwar jederzeit Spieler zu jedem Spiel hinzufügen (siehe Einen Spieler zu einer Spielsitzung hinzufügen), aber mit dem Auffüllen von Matches wird sichergestellt, dass neue Spieler dieselben Spielkriterien erfüllen wie aktuelle Spieler. Darüber hinaus ordnet

das Match-Backfill die neuen Spieler den Teams zu, verwaltet die Spielerakzeptanz und sendet aktualisierte Spielinformationen an den Spieleserver. Weitere Informationen zu Match-Backfill finden Sie in FlexMatchMatchmaking-Prozess.

> **Note**
>
> FlexMatchBackfill ist derzeit nicht für Spiele verfügbar, die Echtzeitserver verwenden.

Es gibt zwei Arten von Backfill-Mechanismen:

- Zur Füllung von Spielesitzungen, die mit weniger als den maximal zulässigen Spielern beginnen; aktivieren Sie hierzu automatisches Backfill.
- Zur Ersetzung von Spielern, die eine laufende Spielesitzung verlassen; ergänzen Sie hierzu Ihren Spieleserver mit Funktionalität zum Senden von Backfill-Anforderungen.

Automatisches Auffüllen einschalten

Beim automatischen Auffüllen von Spielen löst Amazon GameLift automatisch eine Reservierungsanfrage aus, wenn eine Spielsitzung mit einem oder mehreren unbesetzten Spielerplätzen beginnt. Diese Funktion ermöglicht es, Spiele zu starten, sobald die Mindestanzahl passender Spieler zustande gekommen ist, und die verbleibenden Plätze später zu besetzen, wenn zusätzliche Spieler zugeordnet werden. Sie können das automatische Backfill jederzeit stoppen.

Betrachten Sie als Beispiel ein Spiel, das sechs bis zehn Spieler enthalten kann. FlexMatchsucht zunächst sechs Spieler, bildet das Match und startet eine neue Spielsitzung. Beim automatischen Backfill kann die neue Spielesitzung sofort vier weitere Spieler anfordern. Je nach Spielstil möchten wir möglicherweise neuen Spielern die Möglichkeit geben, während der Spielsitzung jederzeit beizutreten. Oder möglicherweise möchten wir das automatische Backfill nach der Ersteinrichtungsphase und vor Beginn des Gameplays stoppen.

Um automatisches Backfill zu Ihrem Spiel hinzufügen zu können, müssen Sie die folgenden Updates an Ihrem Spiel vornehmen.

1. Aktivieren Sie automatisches Backfill. Automatisches Backfill wird in einer Matchmaking-Konfiguration verwaltet. Wenn diese Option aktiviert ist, wird es für alle Matching-Spielsitzungen verwendet, die mit diesem Matchmaker erstellt wurden. Amazon GameLift beginnt mit

der Generierung von Backfill-Anfragen für eine nicht vollständige Spielsitzung, sobald die Spielsitzung auf einem Spieleserver gestartet wird.

Um automatisches Backfill zu aktivieren, öffnen Sie eine Match-Konfiguration und legen Sie als Match-Backfill-Modus „AUTOMATIC“ fest. Weitere Informationen finden Sie unter Erstellen Sie eine Matchmaking-Konfiguration.

2. Schalten Sie die Backfill-Priorisierung ein. Passen Sie Ihren Matchmaking-Prozess an, um das Ausfüllen von Backfill-Anfragen zu priorisieren, bevor Sie neue Matches erstellen. Füge deinem Matchmaking-Regelsatz eine Algorithmuskomponente hinzu und setze die Backfill-Priorität auf „hoch“. Weitere Details finden Sie unter Passen Sie den Match-Algorithmus an.
3. Aktualisieren Sie die Spielsitzung mit neuen Matchmaker-Daten. Amazon GameLift aktualisiert Ihren Spielserver mithilfe der Server-SDK-Callback-Funktion mit Spielinformationen `onUpdateGameSession` (siehe Serverprozess initialisieren). Fügen Sie Ihrem Spieleserver Code hinzu, um aktualisierte Spielsitzungsobjekte als Folge der Backfill-Aktivität verarbeiten zu können. Weitere Informationen finden Sie unter Aktualisiere die Spieldaten auf dem Spielserver.
4. Deaktivieren Sie automatisches Backfill für eine Spielsitzung. Sie können sich jederzeit während einer einzelnen Spielsitzung gegen automatisches Backfill entscheiden. Um das automatische Auffüllen zu beenden, fügen Sie Code zu Ihrem Spielclient oder Spieleserver hinzu, um den GameLift Amazon-API-Aufruf StopMatchmakingdurchzuführen. Dieser Aufruf erfordert eine Ticket-ID. Verwenden Sie die Backfill-Ticket-ID aus der neuesten Backfill-Anforderung. Sie erhalten diese Informationen aus den Spielsitzung-Matchmaking-Daten, die wie im vorherigen Schritt beschrieben aktualisiert werden.

Backfill-Anfragen senden (von einem Spieleserver)

Sie können Match-Backfill-Anforderungen direkt von dem Spielserverprozess aus initiieren, der die Spielsitzung hostet. Der Serverprozess enthält die meisten up-to-date Informationen über aktuelle Spieler, die mit dem Spiel verbunden sind, und den Status leerer Spielerplätze.

In diesem Thema wird davon ausgegangen, dass Sie bereits die notwendigen FlexMatch-Komponenten erstellt und erfolgreich Matchmaking-Prozesse zu Ihrem Spieleserver und einem clientseitigen Spieleservice hinzugefügt haben. Weitere Details zur Einrichtung von FlexMatch finden Sie unter FlexMatchIntegration mit GameLift Amazon-Hosting.

Um Match-Backfill für Ihr Spiel zu ermöglichen, fügen Sie die folgenden Funktionen hinzu:

- Senden Sie Matchmaking-Backfill-Anforderungen an einen Matchmaker und verfolgen Sie den Status der Anfragen.
- Aktualisieren Sie die Spielinformationen für die Spielsitzung. Siehe Aktualisiere die Spieldaten auf dem Spielserver.

Wie bei anderen Serverfunktionen verwendet ein Spieleserver das Amazon GameLift Server SDK. Dieses SDK ist in C++ und C# verfügbar.

Um Match-Backfill-Anforderungen von Ihrem Spielserver zu erstellen, führen Sie die folgenden Aufgaben aus.

1. Lösen Sie eine Match-Backfill-Anforderung aus. Normalerweise wollen Sie eine Backfill-Anforderung auslösen, wenn ein passendes Spiel einen oder mehrere leere Spieler-Slots hat. Sie können Backfill-Anforderungen an bestimmte Umstände knüpfen, z. B. um kritische Rollen zu besetzen oder Teams auszugleichen. Wahrscheinlich möchten Sie auch die Backfilling-Aktivität auf der Grundlage des Alters einer Spielsitzung einschränken.
2. Erstellen Sie eine Backfill-Anforderung. Fügen Sie Code hinzu, um Match-Backfill-Anforderungen zu erstellen und an einen FlexMatch-Matchmaker zu senden. Backfill-Anforderungen werden über diese Server-APIs abgewickelt:

   - StartMatchBackfill()
   - StopMatchBackfill()

   Um eine Backfill-Anforderung zu erstellen, rufen Sie `StartMatchBackfill` mit den folgenden Informationen auf. Um eine Backfill-Anforderung abzubrechen, rufen Sie `StopMatchBackfill` mit der Ticket-ID der Backfill-Anforderung auf.

   - Ticket-ID — Geben Sie eine Matchmaking-Ticket-ID an (oder entscheiden Sie sich dafür, diese automatisch generieren zu lassen). Mit demselben Mechanismus können Sie Ticket-IDs sowohl für Matchmaking-Anforderungen als auch für Backfill-Anforderungen vergeben. Tickets für Matchmaking und Backfilling werden auf die gleiche Weise verarbeitet.
   - Matchmaker — Identifizieren Sie, welcher Matchmaker für die Backfill-Anfrage verwendet werden soll. Im Allgemeinen werden Sie den gleichen Matchmaker verwenden wollen, der auch für die Erstellung des ursprünglichen Matchs verwendet wurde. Diese Anforderung nimmt eine Matchmaking-Konfiguration-ARN entgegen. Diese Informationen werden im Spielesitzungsobjekt (GameSession) gespeichert, das dem Serverprozess von Amazon

GameLift bei der Aktivierung der Spielsitzung zur Verfügung gestellt wurde. Der Matchmaking-Konfiguration-ARN ist in der Eigenschaft `MatchmakerData` enthalten.

- ARN der Spielsitzung — Identifizieren Sie die Spielsitzung, die überfüllt ist. Du kannst den ARN der Spielesitzung abrufen, indem du die Server-API GetGameSessionId() aufrufst. Während des Matchmaking-Prozesses haben Tickets für neue Anfragen keine Spielsitzungs-ID, während Tickets für Backfill-Anforderungen eine solche besitzen. Das Vorhandensein der Sitzungs-ID ist ein Weg, um den Unterschied zwischen Tickets für neue Spiele und Tickets für Backfills zu erkennen.
- Spielerdaten — Füge Spielerinformationen (Spieler) aller aktuellen Spieler in die Spielsitzung ein, die du auffüllen möchtest. Diese Informationen ermöglichen es dem Matchmaker, die bestmöglichen Spiele für die Spieler zu finden, die sich gerade in der Spielsitzung befinden. Sie müssen die Teammitgliedschaft für jeden Spieler angeben. Geben Sie kein Team an, wenn Sie Backfill nicht verwenden. Wenn Ihr Spielserver den Verbindungsstatus des Spielers korrekt gemeldet hat, sollten Sie diese Daten wie folgt erfassen können:
  1. Der Serverprozess, der die Spielsitzung hostet, sollte die meisten up-to-date Informationen darüber enthalten, welche Spieler gerade mit der Spielsitzung verbunden sind.
  2. Um Spieler-IDs, Attribute und Teamzuweisungen abzurufen, rufen Sie Spielerdaten aus dem Spielsitzungsobjekt (GameSession), der `MatchmakerData` Eigenschaft ab (sieheArbeiten Sie mit Matchmaker-Daten). Die Matchmaker-Daten umfassen alle Spieler, die der Spielsitzung zugeordnet wurden, sodass Sie die Spielerdaten nur für die aktuell verbundenen Spieler abrufen müssen.
  3. Sammeln Sie neue Latenzwerte von allen aktuellen Spielern und fügen Sie diese in jedes `Player`-Objekt ein, wenn der Matchmaker Latenzdaten anfordert. Wenn Latenzdaten weggelassen werden und der Matchmaker eine Latenzregel hat, wird die Anfrage nicht erfolgreich zugeordnet. Backfill-Anforderungen erfordern Latenzdaten nur für die Region, in der sich die Spielsitzung gerade befindet. Sie können die Region einer Spielsitzung aus der `GameSessionId`-Eigenschaft des `GameSession`-Objekts abrufen. Dieser Wert ist ein ARN, der die Region enthält.

3. Verfolgen Sie den Status einer Backfill-Anfrage. Amazon GameLift informiert Ihren Gameserver mithilfe der Server-SDK-Callback-Funktion über den Status von Backfill-Anfragen `onUpdateGameSession` (siehe Serverprozess initialisieren). Füge Code für die Verarbeitung der Statusmeldungen — sowie der aktualisierten Spielsitzungsobjekte als Ergebnis erfolgreicher Backfill-Anfragen — unter hinzu. Aktualisiere die Spieldaten auf dem Spielserver

   Ein Matchmaker kann jeweils nur eine Match-Backfill-Anforderung aus einer Spielsitzung verarbeiten. Wenn Sie eine Anfrage stornieren müssen, rufen Sie StopMatchBackfill() an. Wenn

Sie eine Anforderung ändern müssen, rufen Sie `StopMatchBackfill` auf und senden Sie dann eine aktualisierte Anforderung.

Backfill-Anfragen senden (von einem Kundendienst)

Als Alternative zum Senden von Backfill-Anforderungen von einem Spielserver können Sie diese auch von einem clientseitigen Spieleservice aus versenden. Um diese Option zu nutzen, muss der clientseitige Service Zugriff auf aktuelle Daten zur Aktivität der Spielsitzungen und zu den Spielerverbindungen haben. Wenn Ihr Spiel einen Sitzungsverzeichnisdienst verwendet, könnte dies eine gute Wahl sein.

In diesem Thema wird davon ausgegangen, dass Sie bereits die notwendigen FlexMatch-Komponenten erstellt und erfolgreich Matchmaking-Prozesse zu Ihrem Spieleserver und einem clientseitigen Spieleservice hinzugefügt haben. Weitere Details zur Einrichtung von FlexMatch finden Sie unter FlexMatchIntegration mit GameLift Amazon-Hosting.

Um Match-Backfill für Ihr Spiel zu ermöglichen, fügen Sie die folgenden Funktionen hinzu:

- Senden Sie Matchmaking-Backfill-Anforderungen an einen Matchmaker und verfolgen Sie den Status der Anfragen.
- Aktualisieren Sie die Spielinformationen für die Spielsitzung. Siehe Aktualisiere die Spieldaten auf dem Spielserver

Wie bei anderen Client-Funktionen verwendet ein clientseitiger Spieledienst das AWS SDK mit GameLift Amazon-API. Das SDK ist in C++, C # und mehreren anderen Sprachen erhältlich. Eine allgemeine Beschreibung der Client-APIs finden Sie in der Amazon GameLift Service API-Referenz, in der die Low-Level-Service-API für GameLift Amazon-bezogene Aktionen beschrieben wird und Links zu sprachspezifischen Referenzhandbüchern enthält.

Um einen clientseitigen Spieleservice einzurichten, mit dem Sie ein Backfill für Spiele durchführen können, führen Sie die folgenden Aufgaben aus.

1. Lösen Sie eine Anforderung für das Backfilling aus. Normalerweise löst ein Spiel eine Backfill-Anforderung aus, wenn ein passendes Spiel einen oder mehrere leere Spieler-Slots hat. Sie können Backfill-Anforderungen an bestimmte Umstände knüpfen, z. B. um kritische Rollen zu besetzen oder Teams auszugleichen. Wahrscheinlich möchten Sie auch das Backfilling auf der Grundlage des Alters einer Spielsitzung einschränken. Unabhängig vom Auslöser benötigen Sie mindestens die folgenden Informationen. Du kannst diese Informationen aus dem

Spielesitzungsobjekt (GameSession) abrufen, indem du es DescribeGameSessionsmit einer Spielsitzungs-ID aufrufst.

- Anzahl der momentan leeren Spieler-Slots. Dieser Wert kann aus dem maximalen Spielerlimit einer Spielsitzung und der aktuellen Spieleranzahl berechnet werden. Die aktuelle Spieleranzahl wird aktualisiert, wenn Ihr Gameserver den GameLift Amazon-Dienst kontaktiert, um eine Verbindung zu einem neuen Spieler zu überprüfen oder um einen verlassenen Spieler zu melden.

- Creation policy (Erstellungsrichtlinie). Diese Einstellung gibt an, ob die Spielsitzung im Moment neue Spieler akzeptiert.

Das Spielsitzungsobjekt enthält weitere potenziell nützliche Informationen, darunter die Startzeit der Spielsitzung, benutzerdefinierte Spieleigenschaften und Matchmaker-Daten.

2. Erstellen Sie eine Backfill-Anforderung. Fügen Sie Code hinzu, um Match-Backfill-Anforderungen zu erstellen und an einen FlexMatch-Matchmaker zu senden. Backfill-Anforderungen werden über diese Client-APIs abgewickelt:

   - StartMatchBackfill
   - StopMatchmaking

   Um eine Backfill-Anforderung zu erstellen, rufen Sie `StartMatchBackfill` mit den folgenden Informationen auf. Eine Backfill-Anforderung ähnelt einer Matchmaking-Anforderung (siehe Matchmaking für Spieler anfragen), identifiziert aber auch die bestehende Spielsitzung. Um eine Backfill-Anforderung abzubrechen, rufen Sie `StopMatchmaking` mit der Ticket-ID der Backfill-Anforderung auf.

   - Ticket-ID — Geben Sie eine Matchmaking-Ticket-ID an (oder entscheiden Sie sich dafür, diese automatisch generieren zu lassen). Mit demselben Mechanismus können Sie Ticket-IDs sowohl für Matchmaking-Anforderungen als auch für Backfill-Anforderungen vergeben. Tickets für Matchmaking und Backfilling werden auf die gleiche Weise verarbeitet.
   - Matchmaker — Identifizieren Sie den Namen einer zu verwendenden Matchmaking-Konfiguration. Im Allgemeinen werden Sie den gleichen Matchmaker verwenden wollen, der auch für die Erstellung des ursprünglichen Matchs verwendet wurde. Diese Information befindet sich in einem Spielsitzungsobjekt (GameSession), einer `MatchmakerData` Eigenschaft, unter dem Matchmaking-Konfigurations-ARN. Der Namenswert ist die

Zeichenkette, die auf ""matchmakingconfiguration/" folgt. (Im ARN-Wert "arn:aws:gamelift:us-west-2:111122223333:matchmakingconfiguration/MM-4v4" ist der Matchmaking-Configuration-Name z. B. "MM-4v4".)

- Spielesitzungs-ARN — Geben Sie die Spielsitzung an, die aufgefüllt werden soll. Verwenden Sie die `GameSessionId`-Eigenschaft des Spielsitzungsobjekts. Diese ID verwendet den ARN-Wert, den Sie benötigen. Matchmaking-Tickets (MatchmakingTicket) für Backfill-Anfragen haben während der Bearbeitung die Spielsitzungs-ID; Tickets für neue Matchmaking-Anfragen erhalten keine Spielsitzungs-ID, bis das Spiel platziert ist; das Vorhandensein einer Sitzungs-ID im Spiel ist eine Möglichkeit, den Unterschied zwischen Tickets für neue Spiele und Tickets für Backfills zu erkennen.
- Spielerdaten — Füge Spielerinformationen (Spieler) aller aktuellen Spieler in die Spielsitzung ein, die du auffüllen möchtest. Diese Informationen ermöglichen es dem Matchmaker, die bestmöglichen Spiele für die Spieler zu finden, die sich gerade in der Spielsitzung befinden. Sie müssen die Teammitgliedschaft für jeden Spieler angeben. Geben Sie kein Team an, wenn Sie Backfill nicht verwenden. Wenn Ihr Spielserver den Verbindungsstatus des Spielers korrekt gemeldet hat, sollten Sie diese Daten wie folgt erfassen können:

  1. Rufen Sie DescribePlayerSessions() mit der Spielsitzungs-ID auf, um alle Spieler zu ermitteln, die derzeit mit der Spielsitzung verbunden sind. Jede Spielersitzung umfasst eine Spieler-ID. Sie können einen Statusfilter hinzufügen, um nur aktive Spielersitzungen abzurufen.
  2. Ruft Spielerdaten aus dem Spielsitzungsobjekt ab (GameSession), `MatchmakerData` Eigenschaft (sieheArbeiten Sie mit Matchmaker-Daten. Verwenden Sie die im vorherigen Schritt abgerufenen Spieler-IDs, um Daten nur für die aktuell verbundenen Spieler zu erhalten. Da die Matchmaker-Daten beim Ausscheiden von Spielern nicht aktualisiert werden, müssen Sie die Daten der aktuellen Spieler extrahieren.
  3. Sammeln Sie neue Latenzwerte von allen aktuellen Spielern und fügen Sie diese in das `Player`-Objekt ein, wenn der Matchmaker Latenzdaten anfordert. Wenn Latenzdaten weggelassen werden und der Matchmaker eine Latenzregel hat, wird die Anfrage nicht erfolgreich zugeordnet. Backfill-Anforderungen erfordern Latenzdaten nur für die Region, in der sich die Spielsitzung gerade befindet. Sie können die Region einer Spielsitzung aus der `GameSessionId`-Eigenschaft des `GameSession`-Objekts abrufen. Dieser Wert ist ein ARN, der die Region enthält.

3. Verfolgen Sie den Status einer Backfill-Anforderung. Fügen Sie Code hinzu, um den Status von Matchmaking-Tickets zu überprüfen. Sie können den eingerichteten Mechanismus verwenden, um Tickets für neue Matchmaking-Anforderungen zu verfolgen (siehe Verfolgen

Sie Matchmaking-Events), indem Sie die Ereignisbenachrichtigung (empfohlen) oder das Polling nutzen. Obwohl Sie keine Spielerakzeptanz-Aktivität mit Backfill-Anforderungen auslösen müssen und die Spielerinformationen auf dem Spielserver aktualisiert werden, müssen Sie dennoch den Ticketstatus überwachen, um Anforderungsfehler und Neusendungen zu verarbeiten.

Ein Matchmaker kann jeweils nur eine Match-Backfill-Anforderung aus einer Spielsitzung verarbeiten. Wenn Sie eine Anforderung abbrechen möchten, rufen Sie StopMatchmaking auf. Wenn Sie eine Anforderung ändern müssen, rufen Sie `StopMatchmaking` auf und senden Sie dann eine aktualisierte Anforderung.

Sobald eine Match-Backfill-Anforderung erfolgreich ist, erhält Ihr Spielserver ein aktualisiertes `GameSession`-Objekt und übernimmt die Aufgaben, die erforderlich sind, um neue Spieler in die Spielsitzung einzubinden. Weitere Informationen finden Sie unter Aktualisiere die Spieldaten auf dem Spielserver.

Aktualisiere die Spieldaten auf dem Spielserver

Unabhängig davon, wie Sie Backfill-Anfragen für Matches in Ihrem Spiel initiieren, muss Ihr Gameserver in der Lage sein, die Aktualisierungen der Spielsitzung zu verarbeiten, die Amazon aufgrund von Anfragen zum Auffüllen von Matches GameLift bereitstellt.

Wenn Amazon eine Match-Backfill-Anfrage GameLift abschließt — erfolgreich oder nicht —, ruft es Ihren Gameserver mithilfe der Callback-Funktion an. `onUpdateGameSession` Dieser Aufruf hat drei Eingabeparameter: eine Spiel-Backfill-Ticket-ID, eine Statusmeldung und ein GameSession Objekt, das die meisten up-to-date Matchmaking-Daten einschließlich Spielerinformationen enthält. Im Rahmen Ihrer Spielserver-Integration müssen Sie den folgenden Code zu Ihrem Spielserver hinzufügen:

1. Implementieren Sie die `onUpdateGameSession`-Funktion. Diese Funktion muss in der Lage sein, die folgenden Statusmeldungen (`updateReason`) zu verarbeiten:
   - MATCHMAKING_DATA_UPDATED — Neue Spieler wurden erfolgreich der Spielsitzung zugeordnet. Das `GameSession`-Objekt enthält aktualisierte Matchmaker-Daten, einschließlich Spielerdaten zu bestehenden Spielern und neu hinzugekommenen Spielern.
   - BACKFILL_FAILED — Der Versuch, das Spiel aufzufüllen, ist aufgrund eines internen Fehlers fehlgeschlagen. Das `GameSession`-Objekt bleibt unverändert.

- BACKFILL_TIMED_OUT — Der Matchmaker konnte innerhalb des Zeitlimits kein Backfill-Match finden. Das `GameSession`-Objekt bleibt unverändert.
- BACKFILL_CANCELLED — Die Match-Backfill-Anfrage wurde durch einen Aufruf an StopMatchmaking (Client) oder (Server) storniert. StopMatchBackfill Das `GameSession`-Objekt bleibt unverändert.

2. Für erfolgreiche Backfill-Matches verwenden Sie die aktualisierten Matchmaker-Daten, um die neuen Spieler zu verarbeiten, wenn sie sich mit der Spielsitzung verbinden. Sie müssen mindestens die Teamzuweisungen für den/die neuen Spieler sowie andere Spielerattribute verwenden, die erforderlich sind, um den Spieler in das Spiel aufzunehmen.
3. Füge beim Aufruf der Server-SDK-Aktion ProcessReady() deines Gameservers den Namen der `onUpdateGameSession` Callback-Methode als Prozessparameter hinzu.

GameLiftFlexMatchAmazon-Referenz

Dieser Abschnitt enthält Referenzdokumentation für Matchmaking mit Amazon GameLiftFlexMatch.

Themen

- GameLiftFlexMatchAmazon-API-Referenz (AWSSDK)
- FlexMatchRegeln, Sprache
- FlexMatchMatchmaking-Veranstaltungen

GameLiftFlexMatchAmazon-API-Referenz (AWSSDK)

Dieses Thema enthält eine aufgabenbasierte Liste von API-Vorgängen für Amazon. GameLift FlexMatch Die Amazon GameLift FlexMatch Service API ist im AWS SDK im `aws.gamelift` Namespace verpackt. Laden Sie das AWS SDK herunter oder sehen Sie sich die GameLift Amazon-API-Referenzdokumentation an.

Amazon GameLift FlexMatch bietet Matchmaking-Services für Spiele an, die mit GameLift Amazon-Hosting-Lösungen gehostet werden (einschließlich verwaltetem Hosting für benutzerdefinierte Spieleserver oder Echtzeitserver und Hosting auf Amazon EC2 mit Amazon GameLift FleetIQ) sowie mit anderen Hostingsystemen wie lokalen oder peer-to-peer Cloud-Compute-Primitiven. Weitere Informationen zu anderen GameLiftAmazon-Hosting-Optionen finden Sie im Amazon GameLift Developer Guide.

Richten Sie Matchmaking-Regeln und -Prozesse ein

Rufe diese Operationen auf, um einen FlexMatch Matchmaker zu erstellen, den Matchmaking-Prozess für dein Spiel zu konfigurieren und eine Reihe von benutzerdefinierten Regeln für die Erstellung von Matches und Teams zu definieren.

Matchmaking-Konfiguration

- CreateMatchmakingConfiguration— Erstelle eine Matchmaking-Konfiguration mit Anweisungen zur Bewertung von Spielergruppen und zum Aufbau von Spielerteams. Wenn Sie Amazon GameLift für das Hosting verwenden, geben Sie auch an, wie eine neue Spielsitzung für das Spiel erstellt werden soll.
- DescribeMatchmakingConfigurations— Rufen Sie Matchmaking-Konfigurationen ab, die für eine GameLift Amazon-Region definiert wurden.

- UpdateMatchmakingConfiguration— Ändern Sie die Einstellungen für die Matchmaking-Konfiguration. Warteschlange.
- DeleteMatchmakingConfiguration— Entferne eine Matchmaking-Konfiguration aus der Region.

Matchmaking-Regelsatz

- CreateMatchmakingRuleSet— Erstelle eine Reihe von Regeln, die du bei der Suche nach Spielerspielen verwenden kannst.
- DescribeMatchmakingRuleSets— Ruft Matchmaking-Regelsätze ab, die in einer GameLift Amazon-Region definiert wurden.
- ValidateMatchmakingRuleSet— Überprüfen Sie die Syntax für eine Reihe von Matchmaking-Regeln.
- DeleteMatchmakingRuleSet— Entferne einen Matchmaking-Regelsatz aus der Region.

Fordere ein Spiel für einen oder mehrere Spieler an

Rufen Sie diese Vorgänge von Ihrem Spiele-Client-Service an, um Spieler-Matchmaking-Anfragen zu verwalten.

- StartMatchmaking— Fordere Matchmaking für einen Spieler oder eine Gruppe an, die am selben Spiel teilnehmen möchten.
- DescribeMatchmaking— Informieren Sie sich über eine Matchmaking-Anfrage, einschließlich Status.
- AcceptMatch— Bei einem Spiel, für das die Zustimmung eines Spielers erforderlich ist, benachrichtigen Sie Amazon, GameLift wenn ein Spieler ein geplantes Spiel annimmt.
- StopMatchmaking— Stornieren Sie eine Matchmaking-Anfrage.
- StartMatchBackfill- Fordere zusätzliche Spielermatches an, um freie Plätze in einer bestehenden Spielsitzung zu füllen.

Verfügbare Programmiersprachen

Das AWS SDK mit Unterstützung für Amazon GameLift ist in den folgenden Sprachen verfügbar. Informationen zur Unterstützung von Entwicklungsumgebungen finden Sie in der Dokumentation der einzelnen Sprachen.

- C++ (SDK-Dokumente) (Amazon GameLift)
- Java (SDK-Dokumente) (Amazon GameLift)
- .NET (SDK-Dokumente) (Amazon GameLift)
- Go (SDK-Dokumente) (Amazon GameLift)
- Python (SDK-Dokumente) (Amazon GameLift)
- Ruby (SDK-Dokumente) (Amazon GameLift)
- PHP (SDK-Dokumente) (Amazon GameLift)
- JavaScript/Node.js (SDK-Dokumente) (Amazon GameLift)

FlexMatchRegeln, Sprache

Die Referenzthemen in diesem Abschnitt beschreiben die Syntax und Semantik, die zur Erstellung von Matchmaking-Regeln für Amazon verwendet werden. GameLift FlexMatch Ausführliche Hilfe beim Schreiben von Matchmaking-Regeln und Regelsätzen finden Sie unterErstellen Sie einen FlexMatch Regelsatz.

Themen

- FlexMatchRegelsatzschema
- FlexMatchEigenschaftsdefinitionen von Regelsätzen
- FlexMatchRegeltypen
- FlexMatchEigenschaftsausdrücke

FlexMatchRegelsatzschema

FlexMatch-Regelsätze verwenden das Standardschema für Regeln für kleine Matches und große Matches. Eine ausführliche Beschreibung der einzelnen Abschnitte finden Sie unterFlexMatchEigenschaftsdefinitionen von Regelsätzen.

Regelsatzschema für kleine Treffer

Das folgende Schema dokumentiert alle möglichen Eigenschaften und zulässigen Werte für einen Regelsatz, der verwendet wird, um Matches mit bis zu 40 Spielern zu erstellen.

```
{
    "name": "string",
```

```
    "ruleLanguageVersion": "1.0",
    "playerAttributes":[{
         "name": "string,
         "type": <"string", "number", "string_list", "string_number_map">,
         "default": "string"
    }],
    "algorithm": {
        "strategy": "exhaustiveSearch",
        "batchingPreference": <"random", "sorted">,
        "sortByAttributes": [ "string" ],
        "expansionAgeSelection": <"newest", "oldest">,
        "backfillPriority": <"normal", "low", "high">
    },
    "teams": [{
        "name": "string",
        "maxPlayers": number,
        "minPlayers": number,
        "quantity": integer
    }],
    "rules": [{
        "type": "distance",
        "name": "string",
        "description": "string",
        "measurements": "string",
        "referenceValue": number,
        "maxDistance": number,
        "minDistance": number,
        "partyAggregation": <"avg", "min", "max">
        },{
        "type": "comparison",
        "name": "string",
        "description": "string",
        "measurements": "string",
        "referenceValue": number,
        "operation": <"<", "<=", "=", "!=", ">", ">=">,
        "partyAggregation": <"avg", "min", "max">
        },{
        "type": "collection",
        "name": "string",
        "description": "string",
        "measurements": "string",
        "referenceValue": number,
        "operation": <"intersection", "contains", "reference_intersection_count">,
        "maxCount": number,
```

```
        "minCount": number,
        "partyAggregation": <"union", "intersection">
        },{
        "type": "latency",
        "name": "string",
        "description": "string",
        "maxLatency": number,
        "maxDistance": number,
        "distanceReference": number,
        "partyAggregation": <"avg", "min", "max">
        },{
        "type": "distanceSort",
        "name": "string",
        "description": "string",
        "sortDirection": <"ascending", "descending">,
        "sortAttribute": "string",
        "mapKey": <"minValue", "maxValue">,
        "partyAggregation": <"avg", "min", "max">
        },{
        "type": "absoluteSort",
        "name": "string",
        "description": "string",
        "sortDirection": <"ascending", "descending">,
        "sortAttribute": "string",
        "mapKey": <"minValue", "maxValue">,
        "partyAggregation": <"avg", "min", "max">
        },{
        "type": "compound",
        "name": "string",
        "description": "string",
        "statement": "string"
        }
    }],
    "expansions": [{
        "target": "string",
        "steps": [{
            "waitTimeSeconds": number,
            "value": number
        }, {
            "waitTimeSeconds": number,
            "value": number
        }]
    }]
```

```
}
```

Regelsatzschema für große Treffer

Das folgende Schema dokumentiert alle möglichen Eigenschaften und zulässigen Werte für einen Regelsatz, der verwendet wird, um Matches mit mehr als 40 Spielern zu erstellen. Wenn die Gesamtzahl der `maxPlayers` Werte für alle Teams im Regelsatz 40 übersteigt, werden Spielanfragen, die diesen Regelsatz verwenden, gemäß den Richtlinien für große Spiele FlexMatch bearbeitet.

```
{
    "name": "string",
    "ruleLanguageVersion": "1.0",
    "playerAttributes":[{
         "name": "string,
         "type": <"string", "number", "string_list", "string_number_map">,
         "default": "string"
    }],
    "algorithm": {
        "strategy": "balanced",
        "batchingPreference": <"largestPopulation", "fastestRegion">,
        "balancedAttribute": "string",
        "expansionAgeSelection": <"newest", "oldest">,
        "backfillPriority": <"normal", "low", "high">
    },
    "teams": [{
        "name": "string",
        "maxPlayers": number,
        "minPlayers": number,
        "quantity": integer
    }],
    "rules": [{
        "name": "string",
        "type": "latency",
        "description": "string",
        "maxLatency": number,
        "partyAggregation": <"avg", "min", "max">
    }, {
        "name": "string",
        "type": "batchDistance",
        "batchAttribute": "string",
        "maxDistance": number
    }],
```

```
    "expansions": [{
        "target": "string",
        "steps": [{
            "waitTimeSeconds": number,
            "value": number
        }, {
            "waitTimeSeconds": number,
            "value": number
        }]
    }]
}
```

FlexMatchEigenschaftsdefinitionen von Regelsätzen

In diesem Abschnitt wird jede Eigenschaft im Regelsatzschema definiert. Weitere Hilfe beim Erstellen eines Regelsatzes finden Sie unterErstellen Sie einen FlexMatch Regelsatz.

`name`

Eine beschreibende Bezeichnung für den Regelsatz. Dieser Wert ist nicht mit dem Namen verknüpft, der der GameLift MatchmakingRuleSetAmazon-Ressource zugewiesen wurde. Dieser Wert ist in den Matchmaking-Daten enthalten, die ein abgeschlossenes Match beschreiben, wird aber von keinen GameLift Amazon-Prozessen verwendet.

Erlaubte Werte: String

Erforderlich? Nein

`ruleLanguageVersion`

Die Version der verwendeten FlexMatch Eigenschaftsausdruckssprache.

Erlaubte Werte: „1.0"

Erforderlich? Ja

`playerAttributes`

Eine Sammlung von Spielerdaten, die in Matchmaking-Anfragen enthalten sind und im Matchmaking-Prozess verwendet werden. Du kannst hier auch Attribute deklarieren, damit die Spielerdaten in den Matchmaking-Daten enthalten sind, die an die Spielserver weitergegeben werden, auch wenn die Daten nicht für den Matchmaking-Prozess verwendet werden.

Erforderlich? Nein

`name`

Ein eindeutiger Name für das Spielerattribut, das vom Matchmaker verwendet werden soll. Dieser Name muss mit dem Namen des Spielerattributs übereinstimmen, auf das in Matchmaking-Anfragen verwiesen wird.

Erlaubte Werte: String

Erforderlich? Ja

`type`

Der Datentyp des Spieler-Attributwerts.

Erlaubte Werte: „string“, „number“, „string_list“, „string_number_map“

Erforderlich? Ja

`default`

Ein Standardwert, der verwendet wird, wenn eine Matchmaking-Anfrage keinen für einen Spieler bereitstellt.

Zulässige Werte: Jeder zulässige Wert für das Spielerattribut.

Erforderlich? Nein

`algorithm`

Optionale Konfigurationseinstellungen zur Anpassung des Matchmaking-Prozesses.

Erforderlich? Nein

`strategy`

Die Methode, die beim Erstellen von Streichhölzern verwendet werden soll. Wenn diese Eigenschaft nicht gesetzt ist, lautet das Standardverhalten „Erschöpfungssuche“.

Zulässige Werte:

- „Erschöpfende Suche“ — Standardmethode für den Abgleich. FlexMatchbildet anhand des ältesten Tickets in einem Stapel ein Match, indem andere Tickets im Pool anhand benutzerdefinierter Spielregeln bewertet werden. Diese Strategie wird für Spiele mit 40 oder weniger Spielern angewendet. Bei Verwendung dieser Strategie `batchingPreference` sollte entweder auf „zufällig“ oder „sortiert“ gesetzt werden.

- „ausgewogen“ — Methode, die optimiert ist, um schnell große Matches zu bilden. Diese Strategie wird nur für Spiele mit 41 bis 200 Spielern angewendet. Es erstellt Matches, indem es den Ticketpool vorsortiert, potenzielle Spiele erstellt und Spieler Teams zuweist und dann jedes Team in einem Spiel anhand eines bestimmten Spielerattributs ausbalanciert. Diese Strategie kann beispielsweise verwendet werden, um die durchschnittlichen Fähigkeiten aller Teams in einem Spiel auszugleichen. Wenn diese Strategie verwendet wird, `balancedAttribute` muss und `batchingPreference` sollte entweder auf „LargestPopulation“ oder „FastestRegion“ gesetzt werden. Die meisten benutzerdefinierten Regeltypen werden bei dieser Strategie nicht erkannt.

Erforderlich? Ja

`batchingPreference`

Die Methode zur Vorsortierung, die verwendet werden muss, bevor Tickets für das Matchbuilding gruppiert werden. Durch die Vorsortierung des Ticketpools werden die Tickets auf der Grundlage eines bestimmten Merkmals zusammengefasst, wodurch die Einheitlichkeit der Spieler in den Endspielen tendenziell erhöht wird.

Zulässige Werte:

- „random“ — Gültig nur mit `strategy` = „Erschöpfende Suche“. Es erfolgt keine Vorsortierung; die Tickets im Pool werden nach dem Zufallsprinzip gebündelt. Dies ist das Standardverhalten für eine umfassende Suchstrategie.
- „sorted“ — Gültig nur mit `strategy` = „Erschöpfende Suche“. Der Ticketpool ist anhand der unter aufgeführten Spielerattribute vorsortiert. `sortbyAttributes`
- „LargestPopulation“ — Gültig nur mit `strategy` = „balanced“. Der Ticketpool ist vorsortiert nach Regionen, in denen Spieler akzeptable Latenzwerte angeben. Dies ist das Standardverhalten für eine ausgewogene Strategie.
- „FastestRegion“ — Gültig nur mit `strategy` = „balanced“. Der Ticketpool ist vorsortiert nach Regionen, in denen die Spieler ihre niedrigsten Latenzwerte angeben. Das Abschließen von Matches dauert länger, aber die Latenz für alle Spieler ist in der Regel gering.

Erforderlich? Ja

`balancedAttribute`

Der Name eines Spielerattributs, das beim Aufbau großer Matches mit der ausgewogenen Strategie verwendet werden soll.

Zulässige Werte: Jedes Attribut, das `playerAttributes` mit `type` = „Zahl“ deklariert ist.

Erforderlich? Ja, wenn `strategy` = „ausgewogen“.

`sortByAttributes`

Eine Liste von Spielerattributen, die bei der Vorsortierung des Ticketpools vor dem Stapeln verwendet werden sollen. Diese Eigenschaft wird nur bei der Vorsortierung mit der umfassenden Suchstrategie verwendet. Die Reihenfolge der Attributliste bestimmt die Sortierreihenfolge. FlexMatchverwendet die Standardsortierkonvention für Alpha- und Zahlenwerte.

Zulässige Werte: Jedes in deklarierte Attribut`playerAttributes`.

Erforderlich? Ja, if `batchingPreference` = „sortiert“.

`backfillPriority`

Die Priorisierungsmethode für übereinstimmende Backfill-Tickets. Diese Eigenschaft bestimmt, wann die Backfill-Tickets in einem Stapel FlexMatch verarbeitet werden. Es wird nur bei der Vorsortierung mit der umfassenden Suchstrategie verwendet. Wenn diese Eigenschaft nicht gesetzt ist, ist das Standardverhalten „normal“.

Zulässige Werte:

- „normal“ — Die Art der Anfrage eines Tickets (Backfill oder neuer Treffer) wird bei der Bildung von Matches nicht berücksichtigt.
- „hoch“ — Ein Ticketstapel wird nach Anfragetyp (und dann nach Alter) sortiert und FlexMatch versucht zunächst, aufgefüllte Tickets abzugleichen.
- „niedrig“ — Ein Ticketstapel wird nach Anfragetyp (und dann nach Alter) sortiert und FlexMatch versucht zunächst, Tickets ohne Backfill abzugleichen.

Erforderlich? Nein

`expansionAgeSelection`

Die Methode zur Berechnung der Wartezeit für eine Erweiterung der Spielregeln. Erweiterungen werden verwendet, um die Spielanforderungen zu lockern, wenn ein Spiel nach einer bestimmten Zeit nicht abgeschlossen wurde. Die Wartezeit wird anhand des Alters der Tickets berechnet, die sich bereits für das teilweise ausgefüllte Spiel befinden. Wenn diese Eigenschaft nicht gesetzt ist, ist das Standardverhalten „am neuesten“.

Zulässige Werte:

- „neueste“ — Die Wartezeit der Erweiterung wird auf der Grundlage des Tickets mit dem letzten Erstellungszeitstempel des teilweise abgeschlossenen Spiels berechnet.

Erweiterungen werden in der Regel langsamer ausgelöst, da ein neueres Ticket die Wartezeit neu starten kann.

- „älteste“ — Die Wartezeit der Erweiterung wird auf der Grundlage des Tickets mit dem ältesten Erstellungszeitstempel im Spiel berechnet. Erweiterungen werden in der Regel schneller ausgelöst.

Erforderlich? Nein

`teams`

Die Konfiguration der Teams in einem Spiel. Geben Sie für jedes Team einen Teamnamen und einen Größenbereich an. Ein Regelsatz muss mindestens ein Team definieren.

`name`

Ein einzigartiger Name für das Team. Teamnamen können in Regeln und Erweiterungen erwähnt werden. Bei einem erfolgreichen Spiel werden die Spieler in den Matchmaking-Daten nach Teamnamen zugewiesen.

Erlaubte Werte: String

Erforderlich? Ja

`maxPlayers`

Die maximale Anzahl von Spielern, die dem Team zugewiesen werden können.

Zulässige Werte: Zahl

Erforderlich? Ja

`minPlayers`

Die Mindestanzahl an Spielern, die der Mannschaft zugewiesen werden müssen, bevor das Spiel ausgetragen werden kann.

Zulässige Werte: Zahl

Erforderlich? Ja

`quantity`

Die Anzahl der Teams dieses Typs, die in einem Spiel erstellt werden sollen. Teams mit Mengen von mehr als 1 werden mit einer angehängten Zahl gekennzeichnet („Red_1", „Red_2" usw.). Wenn diese Eigenschaft nicht gesetzt ist, ist der Standardwert „1".

Zulässige Werte: Zahl

Erforderlich? Nein

`rules`

Eine Sammlung von Regelbestimmungen, die definieren, wie Spieler für ein Spiel bewertet werden.

Erforderlich? Nein

`name`

Ein eindeutiger Name für die Regel. Alle Regeln in einem Regelsatz müssen eindeutige Namen haben. Regelnamen werden in Ereignisprotokollen und Metriken referenziert, die Aktivitäten im Zusammenhang mit der Regel verfolgen.

Erlaubte Werte: String

Erforderlich? Ja

`description`

Eine Textbeschreibung für die Regel. Diese Informationen können verwendet werden, um den Zweck einer Regel zu identifizieren. Es wird im Matchmaking-Prozess nicht verwendet.

Erlaubte Werte: String

Erforderlich? Nein

`type`

Die Art der Regelaussage. Jeder Regeltyp hat zusätzliche Eigenschaften, die festgelegt werden müssen. Weitere Informationen zur Struktur und Verwendung der einzelnen Regeltypen finden Sie unterFlexMatchRegeltypen.

Zulässige Werte:

- „AbsoluteSort“ — Sortiert mithilfe einer expliziten Sortiermethode, bei der Tickets in einem Stapel danach sortiert werden, ob ein bestimmtes Spielerattribut mit dem ältesten Ticket im Stapel verglichen wird.
- „Sammlung“ — Wertet die Werte in einer Sammlung aus, z. B. ein Spielerattribut, das eine Sammlung ist, oder eine Reihe von Werten für mehrere Spieler.

- „Vergleich“ — Vergleicht zwei Werte.
- „Compound“ — Definiert eine zusammengesetzte Matchmaking-Regel, die eine logische Kombination anderer Regeln im Regelsatz verwendet. Wird nur für Spiele mit 40 oder weniger Spielern unterstützt.
- „Abstand“ — Misst den Abstand zwischen Zahlenwerten.
- „BatchDistance“ — Misst den Unterschied zwischen einem Attributwert und verwendet ihn, um Match-Anfragen zu gruppieren.
- „DistanceSort“ — Sortiert mithilfe einer expliziten Sortiermethode, bei der Tickets in einem Stapel danach sortiert werden, wie ein bestimmtes Spielerattribut mit einem numerischen Wert im Vergleich zum ältesten Ticket im Stapel abschneidet.
- „Latenz“ — Wertet die regionalen Latenzdaten aus, die für eine Matchmaking-Anfrage gemeldet werden.

Erforderlich? Ja

`expansions`

Regeln zur Lockerung der Spielanforderungen im Laufe der Zeit, wenn ein Spiel nicht abgeschlossen werden kann. Richte Erweiterungen als eine Reihe von Schritten ein, die schrittweise angewendet werden, damit Matches leichter zu finden sind. FlexMatchBerechnet standardmäßig die Wartezeit auf der Grundlage des Alters des neuesten Tickets, das einem Spiel hinzugefügt wurde. Sie können mithilfe der Algorithmuseigenschaft ändern, wie die Wartezeiten für Erweiterungen berechnet werden`expansionAgeSelection`.

Bei den Wartezeiten für Erweiterungen handelt es sich um absolute Werte, sodass für jeden Schritt eine längere Wartezeit als für den vorherigen Schritt gelten sollte. Um beispielsweise eine schrittweise Reihe von Erweiterungen zu planen, können Sie Wartezeiten von 30 Sekunden, 40 Sekunden und 50 Sekunden verwenden. Die Wartezeiten dürfen die maximal zulässige Zeit für eine Spielanfrage nicht überschreiten, die in der Matchmaking-Konfiguration festgelegt ist.

Erforderlich? Nein

`target`

Das Regelsatzelement, das gelockert werden soll. Sie können die Eigenschaften für die Teamgröße oder jede Eigenschaft mit Regelanweisungen lockern. Die Syntax lautet "<component name>[<rule/team name>]. <property name>“. Zum Beispiel, um die Mindestgröße von Teams zu ändern:`teams[Red, Yellow].minPlayers`. Um die

Mindestanforderung an Fähigkeiten in einer Vergleichsregel mit dem Namen „minSkill“ zu ändern:`rules[minSkill].referenceValue`.

Erforderlich? Ja

`steps`

`waitTimeSeconds`

Die Wartezeit in Sekunden, bevor der neue Wert auf das Zielregelsatzelement angewendet wird.

Erforderlich? Ja

`value`

Der neue Wert für das Zielregelsatzelement.

FlexMatchRegeltypen

Regel für den Batch-Abstand

```
batchDistance
```

Batch-Entfernungsregeln messen den Unterschied zwischen zwei Attributwerten. Sie können den Regeltyp Batch-Distanz sowohl für große als auch für kleine Übereinstimmungen verwenden. Es gibt zwei Arten von Regeln für den Abstand zwischen Chargen:

- Vergleichen Sie numerische Attributwerte. Eine solche Batch-Distanzregel könnte beispielsweise erfordern, dass alle Spieler in einem Spiel innerhalb von zwei Fähigkeitsstufen voneinander entfernt sind. Definieren Sie für diesen Typ einen maximalen Abstand zwischen dem `batchAttribute` aller Tickets.
- Vergleichen Sie die Werte von Zeichenfolgenattributen. Eine solche Batch-Distanzregel könnte beispielsweise erfordern, dass alle Spieler in einem Spiel denselben Spielmodus anfordern. Definieren Sie für diesen Typ einen `batchAttribute` Wert, der zum Bilden von Stapel FlexMatch verwendet wird.

Eigenschaften der Batch-Abstandsregel

- **`batchAttribute`**— Der Wert des Spielerattributs, der zur Bildung von Stapel verwendet wird.

- **`maxDistance`**— Der maximale Entfernungswert für ein erfolgreiches Spiel. Wird verwendet, um numerische Attribute zu vergleichen.
- **`partyAggregation`**— Der Wert, der bestimmt, wie FlexMatch mit Tickets mit mehreren Spielern (Parteien) umgegangen wird. Zu den gültigen Optionen gehören die Minimal- (`minmax`), Maximal- (`avg`) und Durchschnittswerte () für die Spieler eines Tickets. Der Standardwert ist `avg`.

Example

Beispiele

```
{
  "name":"SimilarSkillRatings",
  "description":"All players must have similar skill ratings",
  "type":"batchDistance",
  "batchAttribute":"SkillRating",
  "maxDistance":"500"
}
```

```
{
  "name":"SameGameMode",
  "description":"All players must have the same game mode",
  "type":"batchDistance",
  "batchAttribute":"GameMode"
}
```

Vergleichsregel

```
comparison
```

In den Vergleichsregeln wird ein Spielerattribut mit einem anderen Wert verglichen. Es gibt zwei Arten von Vergleichsregeln:

- Mit Referenzwert vergleichen. Eine Vergleichsregel dieser Art könnte zum Beispiel verlangen, dass Spieler, die zueinander passen, über ein bestimmtes Fähigkeitsniveau oder höher verfügen. Geben Sie für diesen Typ ein Spielerattribut, einen Referenzwert und eine Vergleichsoperation an.
- Vergleiche zwischen den Spielern. Eine solche Vergleichsregel könnte beispielsweise erfordern, dass alle Spieler im Spiel unterschiedliche Charaktere verwenden. Geben Sie für diesen Typ ein Spielerattribut und entweder die Vergleichsoperation gleich (=) oder ungleich (`!=`) an. Geben Sie keinen Referenzwert an.

> **Note**
>
> Batch-Entfernungsregeln sind effizienter, um Spielerattribute zu vergleichen. Verwenden Sie nach Möglichkeit eine Batch-Entfernungsregel, um die Matchmaking-Latenz zu reduzieren.

Vergleichsregel-Eigenschaften

- **`measurements`**— Der Wert des Spielerattributs, der verglichen werden soll.
- **`referenceValue`**— Der Wert, mit dem die Messung für ein potenzielles Match verglichen werden soll.
- **`operation`**— Der Wert, der bestimmt, wie die Messung mit dem Referenzwert verglichen wird. Gültige Operationen umfassen:`<`,`<=`,`=`,`!=`,`>`,`>=`.
- **`partyAggregation`**— Der Wert, der bestimmt, wie FlexMatch mit Tickets mit mehreren Spielern (Parteien) umgegangen wird. Zu den gültigen Optionen gehören die Minimal- (`minmax`), Maximal- (`avg`) und Durchschnittswerte () für die Spieler eines Tickets. Der Standardwert ist `avg`.

Abstandsregel

```
distance
```

Entfernungsregeln messen den Unterschied zwischen zwei Zahlenwerten, z. B. dem Abstand zwischen den Fähigkeitsstufen der Spieler. Eine Distanzregel könnte beispielsweise erfordern, dass alle Spieler das Spiel mindestens 30 Stunden lang gespielt haben.

> **Note**
>
> Batch-Entfernungsregeln sind effizienter, um Spielerattribute zu vergleichen. Verwenden Sie nach Möglichkeit eine Batch-Entfernungsregel, um die Matchmaking-Latenz zu reduzieren.

Distanzregel-Eigenschaften

- **`measurements`**— Der Wert des Spielerattributs, für das die Entfernung gemessen werden soll. Dies muss ein Attribut mit einem numerischen Wert sein.
- **`referenceValue`**— Der numerische Wert, anhand dessen die Distanz für ein voraussichtliches Spiel gemessen wird.

- **`minDistance`/`maxDistance`**— Der minimale oder maximale Entfernungswert für ein erfolgreiches Spiel.
- **`partyAggregation`**— Der Wert, der bestimmt, wie FlexMatch mit Tickets mit mehreren Spielern (Parteien) umgegangen wird. Zu den gültigen Optionen gehören die Minimal- (`minmax`), Maximal- (`avg`) und Durchschnittswerte () für die Spieler eines Tickets. Der Standardwert ist `avg`.

Erfassungsregel

```
collection
```

Die Sammlungsregeln vergleichen eine Gruppe von Spielerattributwerten mit denen anderer Spieler im Stapel oder mit einem Referenzwert. Eine Sammlung kann Attributwerte für mehrere Spieler, ein Spielerattribut als Zeichenfolgenliste oder beides enthalten. Eine Sammelregel könnte sich beispielsweise mit den Charakteren befassen, die die Spieler in einem Team auswählen. Die Regel könnte dann verlangen, dass das Team mindestens einen bestimmten Charakter hat.

Erfassungsregel-Eigenschaften

- **`measurements`**— Die Sammlung von Spielerattributwerten zum Vergleich. Die Attributwerte müssen Zeichenfolgenlisten sein.
- **`referenceValue`**— Der Wert (oder die Sammlung von Werten), der verwendet werden soll, um Messungen für einen potenziellen Treffer zu vergleichen.
- **`operation`**— Der Wert, der bestimmt, wie eine Sammlung von Messungen verglichen wird. Zu den gültigen Vorgängen gehören die folgenden:
  - `intersection`— Bei dieser Operation wird die Anzahl der Werte gemessen, die in den Sammlungen aller Spieler gleich sind. Ein Beispiel für eine Regel, die die Kreuzungsoperation verwendet, finden Sie unterBeispiel 4: Verwenden Sie die explizite Sortierung, um die besten Übereinstimmungen zu finden.
  - `contains`— Diese Operation misst die Anzahl der Sammlungen von Spielerattributen, die den angegebenen Referenzwert enthalten. Ein Beispiel für eine Regel, die die Operation enthält verwendet, finden Sie unterBeispiel 3: Festlegen von Anforderungen und Latenzlimits auf Teamebene.
  - `reference_intersection_count`— Diese Operation misst die Anzahl der Elemente in einer Spielerattributsammlung, die mit Objekten in der Referenzwertsammlung übereinstimmen. Sie können diese Operation verwenden, um mehrere verschiedene Spielerattribute zu vergleichen.

Ein Beispiel für eine Regel, die Sammlungen mehrerer Spielerattribute vergleicht, finden Sie unter Beispiel 5: Bestimmung von Schnittmengen über mehrere Spielerattribute hinweg.

- **`minCount`/`maxCount`**— Der minimale oder maximale Zählwert für ein erfolgreiches Spiel.
- **`partyAggregation`**— Der Wert, der bestimmt, wie FlexMatch mit Tickets mit mehreren Spielern (Parteien) umgegangen wird. Mit diesem Wert kannst `union` du die Spielerattribute aller Spieler in der Gruppe kombinieren. Sie können auch Spielerattribute verwenden, die die Gruppe gemeinsam hat. `intersection` Der Standardwert ist `union`.

Zusammengesetzte Regel

```
compound
```

Zusammengesetzte Regeln verwenden logische Anweisungen, um Matches mit 40 oder weniger Spielern zu bilden. Sie können mehrere zusammengesetzte Regeln in einem einzigen Regelsatz verwenden. Wenn mehrere zusammengesetzte Regeln verwendet werden, müssen alle zusammengesetzten Regeln wahr sein, damit eine Übereinstimmung entsteht.

Sie können eine zusammengesetzte Regel nicht mithilfe von Erweiterungsregeln erweitern, aber Sie können zugrunde liegende oder unterstützende Regeln erweitern.

Eigenschaften zusammengesetzter Regeln

- **`statement`**— Die Logik, die verwendet wird, um einzelne Regeln zu einer zusammengesetzten Regel zu kombinieren. Die Regeln, die Sie in dieser Eigenschaft angeben, müssen zu einem früheren Zeitpunkt in Ihrem Regelsatz definiert worden sein. Sie können `batchDistance` Regeln nicht in einer zusammengesetzten Regel verwenden.

  Diese Eigenschaft unterstützt die folgenden logischen Operatoren:

  - `and`— Der Ausdruck ist wahr, wenn die beiden angegebenen Argumente wahr sind.
  - `or`— Der Ausdruck ist wahr, wenn eines der beiden angegebenen Argumente wahr ist.
  - `not`— Macht das Ergebnis des Arguments im Ausdruck rückgängig.
  - `xor`— Der Ausdruck ist wahr, wenn nur eines der Argumente wahr ist.

Example Beispiel

Im folgenden Beispiel werden je nach ausgewähltem Spielmodus Spieler mit unterschiedlichen Fähigkeitsstufen zugeordnet.

```
{
    "name": "CompoundRuleExample",
    "type": "compound",
    "statement": "or(and(SeriousPlayers, VeryCloseSkill), and(CasualPlayers,
 SomewhatCloseSkill))"
}
```

Latenz-Regel

```
latency
```

Die Latenzregeln messen die Latenz der Spieler pro Standort. Eine Latenzregel ignoriert jeden Standort mit einer Latenz über dem Maximum. Ein Spieler muss an mindestens einem Ort einen Latenzwert unter dem Maximum haben, damit die Latenzregel ihn akzeptiert. Sie können diesen Regeltyp bei großen Übereinstimmungen verwenden, indem Sie die `maxLatency` Eigenschaft angeben.

Latenzregel-Eigenschaften

- **`maxLatency`**— Der maximal zulässige Latenzwert für einen Standort. Wenn ein Ticket keine Standorte mit einer Latenz unter dem Maximum hat, entspricht das Ticket nicht der Latenzregel.
- **`maxDistance`**— Der maximale Wert zwischen der Latenz jedes Tickets und dem Entfernungsreferenzwert.
- **`distanceReference`**— Der Latenzwert, mit dem die Ticketlatenz verglichen werden soll. Tickets innerhalb der maximalen Entfernung zum Entfernungsreferenzwert führen zu einem erfolgreichen Spiel. Zu den gültigen Optionen gehören die minimalen (`min`) und durchschnittlichen (`avg`) Werte für die Spielerlatenz.
- **`partyAggregation`**— Der Wert, der bestimmt, wie FlexMatch mit Tickets mit mehreren Spielern (Parteien) umgegangen wird. Zu den gültigen Optionen gehören die Minimal- (`minmax`), Maximal- (`avg`) und Durchschnittswerte () für die Spieler eines Tickets. Der Standardwert ist `avg`.

> **Note**
>
> Eine Warteschlange kann eine Spielsitzung in einer Region platzieren, die keiner Latenzregel entspricht. Weitere Informationen zu Latenzrichtlinien für Warteschlangen finden Sie unter Erstellen einer Latenzrichtlinie für Spieler.

Absolute Sortierregel

```
absoluteSort
```

Absolute Sortierregeln sortieren einen Stapel von Matchmaking-Tickets anhand eines bestimmten Spielerattributs im Vergleich zum ersten Ticket, das dem Stapel hinzugefügt wurde.

Absolutsortierungsregel-Eigenschaften

- **`sortDirection`**— Die Reihenfolge, in der die Matchmaking-Tickets sortiert werden. Zu den gültigen Optionen gehören `ascending` und`descending`.
- **`sortAttribute`**— Das Spielerattribut, nach dem Tickets sortiert werden sollen.
- **`mapKey`**— Die Optionen zum Sortieren des Spielerattributs, wenn es sich um eine Map handelt. Gültige Optionen sind unter anderem:
  - `minValue`— Der Schlüssel mit dem niedrigsten Wert steht an erster Stelle.
  - `maxValue`— Der Schlüssel mit dem höchsten Wert steht an erster Stelle.
- **`partyAggregation`**— Der Wert, der bestimmt, wie FlexMatch mit Tickets mit mehreren Spielern (Parteien) umgegangen wird. Zu den gültigen Optionen gehören das minimale Spielerattribut (`min`), das maximale (`max`) Spielerattribut und der Durchschnitt (`avg`) aller Spielerattribute für Spieler in der Gruppe. Der Standardwert ist `avg`.

Example

Beispiel

In der folgenden Beispielregel werden die Spieler nach ihrer Fähigkeitsstufe sortiert und der Durchschnittswert der Spielstärke der Gruppen ermittelt.

```
{
    "name":"AbsoluteSortExample",
    "type":"absoluteSort",
    "sortDirection":"ascending",
    "sortAttribute":"skill",
    "partyAggregation":"avg"
}
```

Regel zum Sortieren von Entfernungen

```
distanceSort
```

Die Regeln für die Entfernungssortierung sortieren einen Stapel von Matchmaking-Tickets anhand der Entfernung eines bestimmten Spielerattributs vom ersten Ticket, das dem Stapel hinzugefügt wurde.

Distanzsortierungsregel-Eigenschaften

- **`sortDirection`**— Die Richtung zum Sortieren von Matchmaking-Tickets. Zu den gültigen Optionen gehören `ascending` und`descending`.
- **`sortAttribute`**— Das Spielerattribut, nach dem Tickets sortiert werden sollen.
- **`mapKey`**— Die Optionen zum Sortieren des Spielerattributs, wenn es sich um eine Map handelt. Gültige Optionen sind unter anderem:
  - `minValue`— Suchen Sie für das erste Ticket, das dem Stapel hinzugefügt wurde, den Schlüssel mit dem niedrigsten Wert.
  - `maxValue`— Suchen Sie für das erste Ticket, das dem Stapel hinzugefügt wurde, den Schlüssel mit dem höchsten Wert.
- **`partyAggregation`**— Der Wert, der bestimmt, wie FlexMatch mit Tickets mit mehreren Spielern (Parteien) umgegangen wird. Zu den gültigen Optionen gehören die Minimal- (`minmax`), Maximal- (`avg`) und Durchschnittswerte () für die Spieler eines Tickets. Der Standardwert ist `avg`.

FlexMatchEigenschaftsausdrücke

Eigenschaftsausdrücke können verwendet werden, um bestimmte Eigenschaften im Zusammenhang mit Matchmaking zu definieren. Sie ermöglichen es Ihnen, bei der Definition eines Eigenschaftswerts Berechnungen und Logik zu verwenden. Eigenschaftsausdrücke führen im Allgemeinen zu einer von zwei Formen:

- Individuelle Spielerdaten.
- Berechnete Sammlungen von individuellen Spielerdaten.

Allgemeine Ausdrücke für Matchmaking-Eigenschaften

Ein Eigenschaftsausdruck identifiziert einen bestimmten Wert für einen Spieler, eine Mannschaft oder ein Spiel. Die folgenden teilweisen Ausdrücke veranschaulichen, wie Teams und Spieler identifiziert werden:

Ziel	Eingabe	Bedeutung	Ausgabe
Identifizierung eines bestimmten Teams in einem Match:	`teams[red]`	Das rote Team	Team
Um eine Gruppe bestimmter Teams in einem Spiel zu identifizieren:	`teams[red,blue]`	Das rote Team und das blaue Team	List<Team>
Identifizierung aller Teams in einem Match:	`teams[*]`	Alle Teams	List<Team>
Identifizierung von Spielern in einem bestimmten Team:	`team[red] .players`	Spieler im roten Team	List<Player>
Um Spieler einer Gruppe bestimmter Teams in einem Spiel zu identifizieren:	`team[red, blue].players`	Spieler im Match, gruppiert nach Team	List<List<Player>>
Identifizierung von Spielern in einem Match:	`team[*].players`	Spieler im Match, gruppiert nach Team	List<List<Player>>

Beispiele für Eigenschaftsausdrücke

Die folgende Tabelle zeigt einige Eigenschaftsausdrücke, die auf den vorherigen Beispielen aufbauen:

Expression	Bedeutung	Ergebnistyp
`teams[red].players[playerId]`	Die Spieler-IDs aller Spieler aus dem roten Team	List<string>
`teams[red].players.attributes[skill]`	Die „skill“-Attribute aller Spieler aus dem roten Team	List<number>
`teams[red,blue].players.attributes[skill]`	Die „Fähigkeiten“ aller Spieler der roten und der blauen Mannschaft, gruppiert nach Teams	List<List<number>>
`teams[*].players.attributes[skill]`	Die „skill“-Attribute aller Spieler im Match, gruppiert nach Team	List<List<number>>

Aggregationen von Eigenschaftsausdrücken

Eigenschaftsausdrücke können verwendet werden, um Teamdaten unter Verwendung der folgenden Funktionen oder Kombinationen von Funktionen zu aggregieren:

Aggregation	Eingabe	Bedeutung	Ausgabe
`min`	List<number>	Minimum aller Zahlen in der Liste ermitteln.	number (Zahl
`max`	List<number>	Maximum aller Zahlen in der Liste ermitteln.	number (Zahl
`avg`	List<number>	Durchschnitt aller Zahlen in der Liste ermitteln.	number (Zahl
`median`	List<number>	Median aller Zahlen in der Liste ermitteln.	number (Zahl

Aggregation	Eingabe	Bedeutung	Ausgabe
`sum`	List<number>	Summe aller Zahlen in der Liste ermitteln.	number (Zahl
`count`	List<?>	Anzahl aller Elemente in der Liste ermitteln.	number (Zahl
`stddev`	List<number>	Standardabweichung aller Zahlen in der Liste ermitteln.	number (Zahl
`flatten`	List<List<?>>	eine Sammlung verschachtelter Listen in eine Liste mit allen Elementen umwandeln.	List<?>
`set_intersection`	List<List<string>>	Ermittelt eine Liste von Zeichenfolgen, die in allen Zeichenfolge-Listen in einer Sammlung enthalten sind.	List<string>
All above	List<List<?>>	Alle Operationen für eine verschachtelte Liste bearbeiten jede Unterliste individuell, um eine Ergebnisliste zu erstellen.	List<?>

Die folgende Tabelle zeigt einige gültige Eigenschaftsausdrücke, die Aggregationsfunktionen verwenden:

Expression	Bedeutung	Ergebnistyp
flatten(teams[*].players.attributes[skill])	Die „skill“-Attribute aller Spieler im Match (nicht gruppiert)	List<number>
avg(teams[red].players.attributes[skill])	Die durchschnittliche Qualifikation der Spieler aus dem roten Team	number (Zahl
avg (teams [*] .players.attributes [Fähigkeit])	Die durchschnittliche Qualifikation jedes Teams im Match	List<number>
avg(flatten(teams[*].players.attributes[skill]))	Das durchschnittliche Qualifikationslevel aller Spieler im Match. Dieser Ausdruck erhält eine flache Liste der Spielerqu alifikationen und erzeugt dann die Mittelwerte dafür.	number (Zahl
count(teams[red].players)	Die Anzahl der Spieler aus dem roten Team	number (Zahl
count (teams[*].players)	Die Anzahl der Spieler jedes Teams im Match	List<number>
max(avg(teams[*].players.attributes[skill]))	Die höchste Teamqualifikation im Match	number (Zahl

FlexMatchMatchmaking-Veranstaltungen

Amazon GameLift FlexMatch sendet Ereignisse für jedes Matchmaking-Ticket, während es verarbeitet wird. Sie können diese Ereignisse zu einem Amazon SNS-Thema veröffentlichen, wie unter beschriebenFlexMatchEventbenachrichtigungen einrichten. Diese Ereignisse werden außerdem nahezu in Echtzeit und auf Best-Effort-Basis an Amazon CloudWatch Events gesendet.

Dieses Thema beschreibt die Struktur von FlexMatch Ereignissen und enthält ein Beispiel für jeden Ereignistyp. Weitere Informationen zum Status von Matchmaking-Tickets finden Sie MatchmakingTicketin der Amazon GameLift API-Referenz.

MatchmakingSearching

Das Ticket wurde in Matchmaking eingegeben. Dies umfasst neue Anforderungen und Anforderungen, die Teil eines fehlgeschlagenen vorgeschlagenen Match sind.

Ressource: ConfigurationArn

Detail: Typ, Tickets,estimatedWaitMillis, gameSessionInfo

Beispiel

```
{
  "version": "0",
  "id": "cc3d3ebe-1d90-48f8-b268-c96655b8f013",
  "detail-type": "GameLift Matchmaking Event",
  "source": "aws.gamelift",
  "account": "123456789012",
  "time": "2017-08-08T21:15:36.421Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:gamelift:us-west-2:123456789012:matchmakingconfiguration/
SampleConfiguration"
  ],
  "detail": {
    "tickets": [
      {
        "ticketId": "ticket-1",
        "startTime": "2017-08-08T21:15:35.676Z",
        "players": [
          {
            "playerId": "player-1"
```

```
          }
        ]
      }
    ],
    "estimatedWaitMillis": "NOT_AVAILABLE",
    "type": "MatchmakingSearching",
    "gameSessionInfo": {
      "players": [
        {
          "playerId": "player-1"
        }
      ]
    }
  }
}
```

PotentialMatchCreated

Ein potenzielles Match wurde erstellt. Dies wird für alle neuen potenziellen Übereinstimmungen ausgegeben, unabhängig davon, ob die Annahme erforderlich ist.

Ressource: ConfigurationArn

Detail: Typ, Tickets, AcceptanceTimeout, AcceptanceRequired,,, MatchID ruleEvaluationMetrics gameSessionInfo

Beispiel

```
{
  "version": "0",
  "id": "fce8633f-aea3-45bc-aeba-99d639cad2d4",
  "detail-type": "GameLift Matchmaking Event",
  "source": "aws.gamelift",
  "account": "123456789012",
  "time": "2017-08-08T21:17:41.178Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:gamelift:us-west-2:123456789012:matchmakingconfiguration/
SampleConfiguration"
  ],
  "detail": {
    "tickets": [
      {
```

```
        "ticketId": "ticket-1",
        "startTime": "2017-08-08T21:15:35.676Z",
        "players": [
          {
            "playerId": "player-1",
            "team": "red"
          }
        ]
      },
      {
        "ticketId": "ticket-2",
        "startTime": "2017-08-08T21:17:40.657Z",
        "players": [
          {
            "playerId": "player-2",
            "team": "blue"
          }
        ]
      }
    ],
    "acceptanceTimeout": 600,
    "ruleEvaluationMetrics": [
      {
        "ruleName": "EvenSkill",
        "passedCount": 3,
        "failedCount": 0
      },
      {
        "ruleName": "EvenTeams",
        "passedCount": 3,
        "failedCount": 0
      },
      {
        "ruleName": "FastConnection",
        "passedCount": 3,
        "failedCount": 0
      },
      {
        "ruleName": "NoobSegregation",
        "passedCount": 3,
        "failedCount": 0
      }
    ],
    "acceptanceRequired": true,
```

```
    "type": "PotentialMatchCreated",
    "gameSessionInfo": {
      "players": [
        {
          "playerId": "player-1",
          "team": "red"
        },
        {
          "playerId": "player-2",
          "team": "blue"
        }
      ]
    },
    "matchId": "3faf26ac-f06e-43e5-8d86-08feff26f692"
  }
}
```

AcceptMatch

Spieler haben ein potenzielles Match akzeptiert. Dieses Ereignis enthält den aktuellen Akzeptanzstatus jedes Spielers im Match. Fehlende Daten bedeuten, AcceptMatch dass das für diesen Spieler nicht aufgerufen wurde.

Ressource: ConfigurationArn

Detail: Typ, Tickets, MatchID, gameSessionInfo

Beispiel

```
{
  "version": "0",
  "id": "b3f76d66-c8e5-416a-aa4c-aa1278153edc",
  "detail-type": "GameLift Matchmaking Event",
  "source": "aws.gamelift",
  "account": "123456789012",
  "time": "2017-08-09T20:04:42.660Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:gamelift:us-west-2:123456789012:matchmakingconfiguration/
SampleConfiguration"
  ],
  "detail": {
    "tickets": [
```

```
      {
        "ticketId": "ticket-1",
        "startTime": "2017-08-09T20:01:35.305Z",
        "players": [
          {
            "playerId": "player-1",
            "team": "red"
          }
        ]
      },
      {
        "ticketId": "ticket-2",
        "startTime": "2017-08-09T20:04:16.637Z",
        "players": [
          {
            "playerId": "player-2",
            "team": "blue",
            "accepted": false
          }
        ]
      }
    ],
    "type": "AcceptMatch",
    "gameSessionInfo": {
      "players": [
        {
          "playerId": "player-1",
          "team": "red"
        },
        {
          "playerId": "player-2",
          "team": "blue",
          "accepted": false
        }
      ]
    },
    "matchId": "848b5f1f-0460-488e-8631-2960934d13e5"
  }
}
```

AcceptMatchCompleted

Die Match-Akzeptanz ist durch Spieler-Akzeptanz, Spieler-Ablehnung oder Akzeptanz-Timeout erfolgt.

Ressource: ConfigurationArn

Detail: Typ, Tickets, Annahme, MatchID, gameSessionInfo

Beispiel

```
{
  "version": "0",
  "id": "b1990d3d-f737-4d6c-b150-af5ace8c35d3",
  "detail-type": "GameLift Matchmaking Event",
  "source": "aws.gamelift",
  "account": "123456789012",
  "time": "2017-08-08T20:43:14.621Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:gamelift:us-west-2:123456789012:matchmakingconfiguration/
SampleConfiguration"
  ],
  "detail": {
    "tickets": [
      {
        "ticketId": "ticket-1",
        "startTime": "2017-08-08T20:30:40.972Z",
        "players": [
          {
            "playerId": "player-1",
            "team": "red"
          }
        ]
      },
      {
        "ticketId": "ticket-2",
        "startTime": "2017-08-08T20:33:14.111Z",
        "players": [
          {
            "playerId": "player-2",
            "team": "blue"
          }
```

```
        ]
      }
    ],
    "acceptance": "TimedOut",
    "type": "AcceptMatchCompleted",
    "gameSessionInfo": {
      "players": [
        {
          "playerId": "player-1",
          "team": "red"
        },
        {
          "playerId": "player-2",
          "team": "blue"
        }
      ]
    },
    "matchId": "a0d9bd24-4695-4f12-876f-ea6386dd6dce"
  }
}
```

MatchmakingSucceeded

Das Matchmaking wurde erfolgreich abgeschlossen und eine Spielsitzung wurde erstellt.

Ressource: ConfigurationArn

Detail: Typ, Tickets, MatchID, gameSessionInfo

Beispiel

```
{
  "version": "0",
  "id": "5ccb6523-0566-412d-b63c-1569e00d023d",
  "detail-type": "GameLift Matchmaking Event",
  "source": "aws.gamelift",
  "account": "123456789012",
  "time": "2017-08-09T19:59:09.159Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:gamelift:us-west-2:123456789012:matchmakingconfiguration/
SampleConfiguration"
  ],
```

```
  "detail": {
    "tickets": [
      {
        "ticketId": "ticket-1",
        "startTime": "2017-08-09T19:58:59.277Z",
        "players": [
          {
            "playerId": "player-1",
            "playerSessionId": "psess-6e7c13cf-10d6-4756-a53f-db7de782ed67",
            "team": "red"
          }
        ]
      },
      {
        "ticketId": "ticket-2",
        "startTime": "2017-08-09T19:59:08.663Z",
        "players": [
          {
            "playerId": "player-2",
            "playerSessionId": "psess-786b342f-9c94-44eb-bb9e-c1de46c472ce",
            "team": "blue"
          }
        ]
      }
    ],
    "type": "MatchmakingSucceeded",
    "gameSessionInfo": {
      "gameSessionArn": "arn:aws:gamelift:us-west-2:123456789012:gamesession/836cf48d-
bcb0-4a2c-bec1-9c456541352a",
      "ipAddress": "192.168.1.1",
      "port": 10777,
      "players": [
        {
          "playerId": "player-1",
          "playerSessionId": "psess-6e7c13cf-10d6-4756-a53f-db7de782ed67",
          "team": "red"
        },
        {
          "playerId": "player-2",
          "playerSessionId": "psess-786b342f-9c94-44eb-bb9e-c1de46c472ce",
          "team": "blue"
        }
      ]
    },
```

```
    "matchId": "c0ec1a54-7fec-4b55-8583-76d67adb7754"
  }
}
```

MatchmakingTimedOut

Das Matchmaking-Ticket ist aufgrund einer Zeitüberschreitung fehlgeschlagen.

Ressource: ConfigurationArn

Detail: Typ, TicketsruleEvaluationMetrics, Nachricht, MatchID, gameSessionInfo

Beispiel

```
{
  "version": "0",
  "id": "fe528a7d-46ad-4bdc-96cb-b094b5f6bf56",
  "detail-type": "GameLift Matchmaking Event",
  "source": "aws.gamelift",
  "account": "123456789012",
  "time": "2017-08-09T20:11:35.598Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:gamelift:us-west-2:123456789012:matchmakingconfiguration/
SampleConfiguration"
  ],
  "detail": {
    "reason": "TimedOut",
    "tickets": [
      {
        "ticketId": "ticket-1",
        "startTime": "2017-08-09T20:01:35.305Z",
        "players": [
          {
            "playerId": "player-1",
            "team": "red"
          }
        ]
      }
    ],
    "ruleEvaluationMetrics": [
      {
        "ruleName": "EvenSkill",
```

```
        "passedCount": 3,
        "failedCount": 0
      },
      {
        "ruleName": "EvenTeams",
        "passedCount": 3,
        "failedCount": 0
      },
      {
        "ruleName": "FastConnection",
        "passedCount": 3,
        "failedCount": 0
      },
      {
        "ruleName": "NoobSegregation",
        "passedCount": 3,
        "failedCount": 0
      }
    ],
    "type": "MatchmakingTimedOut",
    "message": "Removed from matchmaking due to timing out.",
    "gameSessionInfo": {
      "players": [
        {
          "playerId": "player-1",
          "team": "red"
        }
      ]
    }
  }
}
```

MatchmakingCancelled

Das Matchmaking-Ticket wurde storniert.

Ressource: ConfigurationArn

Detail: Typ, TicketsruleEvaluationMetrics, Nachricht, MatchID, gameSessionInfo

Beispiel

```
{
```

```
  "version": "0",
  "id": "8d6f84da-5e15-4741-8d5c-5ac99091c27f",
  "detail-type": "GameLift Matchmaking Event",
  "source": "aws.gamelift",
  "account": "123456789012",
  "time": "2017-08-09T20:00:07.843Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:gamelift:us-west-2:123456789012:matchmakingconfiguration/
SampleConfiguration"
  ],
  "detail": {
    "reason": "Cancelled",
    "tickets": [
      {
        "ticketId": "ticket-1",
        "startTime": "2017-08-09T19:59:26.118Z",
        "players": [
          {
            "playerId": "player-1"
          }
        ]
      }
    ],
    "ruleEvaluationMetrics": [
      {
        "ruleName": "EvenSkill",
        "passedCount": 0,
        "failedCount": 0
      },
      {
        "ruleName": "EvenTeams",
        "passedCount": 0,
        "failedCount": 0
      },
      {
        "ruleName": "FastConnection",
        "passedCount": 0,
        "failedCount": 0
      },
      {
        "ruleName": "NoobSegregation",
        "passedCount": 0,
        "failedCount": 0
```

```
      }
    ],
    "type": "MatchmakingCancelled",
    "message": "Cancelled by request.",
    "gameSessionInfo": {
      "players": [
        {
          "playerId": "player-1"
        }
      ]
    }
  }
}
```

MatchmakingFailed

Für das Matchmaking-Ticket ist ein Fehler aufgetreten. Der Grund kann sein, dass die Spielsitzungswarteschlange nicht verfügbar ist, oder dass ein interner Fehler aufgetreten ist.

Ressource: ConfigurationArn

Detail: Typ, TicketsruleEvaluationMetrics, Nachricht, MatchID, gameSessionInfo

Beispiel

```
{
  "version": "0",
  "id": "025b55a4-41ac-4cf4-89d1-f2b3c6fd8f9d",
  "detail-type": "GameLift Matchmaking Event",
  "source": "aws.gamelift",
  "account": "123456789012",
  "time": "2017-08-16T18:41:09.970Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:gamelift:us-west-2:123456789012:matchmakingconfiguration/
SampleConfiguration"
  ],
  "detail": {
    "tickets": [
      {
        "ticketId": "ticket-1",
        "startTime": "2017-08-16T18:41:02.631Z",
        "players": [
```

```
          {
            "playerId": "player-1",
            "team": "red"
          }
        ]
      }
    ],
    "customEventData": "foo",
    "type": "MatchmakingFailed",
    "reason": "UNEXPECTED_ERROR",
    "message": "An unexpected error was encountered during match placing.",
    "gameSessionInfo": {
      "players": [
        {
          "playerId": "player-1",
          "team": "red"
        }
      ]
    },
    "matchId": "3ea83c13-218b-43a3-936e-135cc570cba7"
  }
}
```

Sicherheit in FlexMatch

Die Sicherheit in der Cloud hat bei AWS höchste Priorität. Als AWS-Kunde profitieren Sie von Rechenzentren und Netzwerkarchitekturen, die eingerichtet wurden, um die Anforderungen der anspruchsvollsten Organisationen in puncto Sicherheit zu erfüllen.

Sicherheit gilt zwischen AWS und Ihnen eine geteilte Verantwortung. Informationen zur Anwendung des Modells der geteilten Verantwortung bei der Verwendung FlexMatch finden Sie unter Sicherheit bei Amazon GameLift.

GameLiftFlexMatchAmazon-Versionshinweise und SDK-Versionen

Die GameLift Versionshinweise von Amazon enthalten Details zu neuen FlexMatch Funktionen, Updates und Korrekturen im Zusammenhang mit dem Service. Diese Seite enthält auch den Amazon GameLift SDK-Versionsverlauf.

Ressourcen für GameLift Amazon-Entwickler

Die gesamte GameLift Amazon-Dokumentation und Entwicklerressourcen finden Sie auf der GameLiftAmazon-Dokumentations-Homepage.

AWS-Glossar

Die neueste AWS-Terminologie finden Sie im AWS-Glossar in der AWS-Glossar-Referenz.

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.