Game Player Matchmaking

Development Guide

Product Documentation

Tencent Cloud

Service Statement

This document is intended to provide users with general information about Tencent Cloud's products and services only and does not form part of Tencent Cloud's terms and conditions. Tencent Cloud's products or services are subject to change. Specific products and services and the standards applicable to them are exclusively provided for in Tencent Cloud's applicable terms and conditions.

Contents

Development Guide

Match Implementation Mechanism

MatchTicket Parameters

Last updated : 2022-03-29 18:51:51

> Note :
> Since the product logic no longer meets the technical development of game industry, Game Player Matching GPM will be deprecated on June 1st, 2022. Please complete service migration before May 31 , 2022.

This document describes GPM's MatchTicket parameters. For more information about relevant concepts, see Glossary.

Status

MatchTicket status changes during the lifecycle of a matchmaking request. The updated status will be recorded in the **Status** parameter.

MatchTicket statuses

- **SEARCHING**: MatchTicket in this status indicates that GPM is searching for other satisfactory tickets in the match pool to be matched to this MatchTicket within the timeout period. If no ticket is found during this period, the MatchTicket status will change to **TIMEDOUT**.
- **PLACING**: MatchTicket in this status indicates that GPM is placing the matchmaking result of a complete match to the Tencent Cloud Game Server Engine (GSE) to start a new game server session in GSE. Only MatchTickets with a MatchCode configured to request resources from GSE will be in this status.
- **COMPLETED**: MatchTicket in this status indicates that a satisfactory match is completed, and its lifecycle in GPM ends.
    - For a MatchTicket without requesting battle server resources, this status indicates that the matchmaking request is completed.
    - For a MatchTicket automatically requiring battle server resources from GSE, this status indicates that the matchmaking request is completed and the matchmaking result is placed to the specified GSE server queue.

- **CANCELLED**: MatchTicket in this status means that the user cancels the matchmaking request at the **SEARCHING** stage.
- **TIMEDOUT**: MatchTicket in this status indicates that no satisfactory tickets are found for the matchmaking request within the specified timeout period at the **SEARCHING** stage. TimeOut is a MatchCode parameter that is defined when a match is created.
- **FAILED**: MatchTicket in this status indicate that the matchmaking request fails due to internal error at the **SEARCHING or PLACING** stage.

MatchResult Fields

- If the value of `MatchType` **is** `NORMAL` **, the MatchResult fields should be parsed as follows**:
  - **Parameters**

Field	Type	Description
MatchedPlayers	Array of MatchedPlayer	Players matched to a game battle

  - **Sample code**

```
"MatchResult": "{¥"MatchedPlayers¥":[{¥"PlayerId¥":¥"xxxx¥", ¥"PlayerSessionId¥":¥"¥", ¥"MatchTicketId¥":¥"xxxxxxx¥"}, {¥"PlayerId¥":¥"xxxxx¥", ¥"PlayerSessionId¥":¥"¥", ¥"MatchTicketId¥":¥"xxxxxxx¥"}]}"
```

- If the value of `MatchType` **is** `GSE` **, the MatchResult fields should be parsed as follows**:
  - **Parameters**

Field	Type	Description
MatchedPlayers	Array of MatchedPlayer	Players matched to a game battle
DnsName	string	DNS flag returned from GSE. This field may be empty.
GameServerSessionId	string	Game server session ID returned from GSE
IpAddress	string	IP address returned from GSE
Port	number	Port number returned from GSE

  - **Sample code**

```
”MatchResult”: ”{¥”DnsName¥”:¥”¥”, ¥”GameServerSessionId¥”:¥”:gameserversession/fleet-xxx¥”,
¥”IpAddress¥”:¥”xx.xx.xx.xx¥”, ¥”Port¥”:xxx, ¥”MatchedPlayers¥”:[{¥”PlayerId¥”:¥”xxx¥”, ¥”Pl
ayerSessionId¥”:¥”psess-xxxxxx¥”, ¥”MatchTicketId¥”:¥”xxx¥”}, {¥”PlayerId¥”:¥”xxx¥”, ¥”Playe
rSessionId¥”:¥”psess-xxxxxxx¥”, ¥”MatchTicketId¥”:¥”xxxxxxx¥”}]}”
```

- **MatchedPlayer**

Field	Type	Description
PlayerId	string	Player ID, which is passed in when a match is initiated.
PlayerSessionId	string	Player session ID returned from GSE. This field is empty if the value of MatchType is `NORMAL`.
MatchTicketId	string	MatchTicket ID of the current players.

Multi-player Match

Last updated : 2022-03-29 18:51:51

> Note :
> Since the product logic no longer meets the technical development of game industry, Game Player Matching GPM will be deprecated on June 1st, 2022. Please complete service migration before May 31 , 2022.

The document describes the GPM team match mechanism.

Initiating Team Match

Players passed in the StartMatching API initiate a team match. These players have the same MatchTicket.

GPM rule on team match ticket

- **partyAggregation**
  Configure the `partyAggregation` parameter in the rule script to define the calculations of team player attributes.

Valid Value	Description	Applied Parameters
avg	Each player owning the same MatchTicket will use the average team attribute.	number and latency
min	Each player owning the same MatchTicket will use the minimum attribute value of the team.	number and latency
max	Each player owning the same MatchTicket will use the maximum attribute value of the team.	number and latency
each	Each player owning the same MatchTicket will use their respective attribute value.	string, number, and latency
any	Each player owning the same MatchTicket will use the same random attribute value of the team.	string, number, and latency

> Note :
> The `partyAggregation` parameter defined in the player attribute rule only takes effect on players owning the same MatchTicket that specify the same team.

Examples

See below for a sample rule configuration:

```
{
”version”: ”v1.0”,
”playerAttributes”: [{
”name”: ”skill”,
”type”: ”number”
}],
”teams”: [{
”name”: ”red”,
”maxPlayers”: 4,
”minPlayers”: 4
}],
”rules”: [ {
”name”: ”TeamSkill”,
”measurements”: [
”avg(teams[red].players.playerAttributes[skill])”
],
”referenceValue”: 30,
”maxDistance”: 10,
”partyAggregation”:”max”
}]
}
```

This rule shows that four players need to be matched to the red team, and the difference between the TeamSkill average and the reference value 30 should be less than 10. The `partyAggregation` parameter is configured to “max”, indicating that the each player owning the same MatchTicket should use the maximum attribute value of the team, as shown below:

- Assume MatchTicket1 applies to three team match players: player_A, player_B, and player_C.
- Assume MatchTicket2 applies to one team match player: player_D.
- The skill attributes of the four players are: A (20), B (30), C (35), and D (30).

As shown above, max (A, B, C) = 35, and max (D) = 30. To determine whether MatchTicket1 and MatchTicket 2 satisfy the matchmaking conditions of the TeamSkill rule, GPM firstly replace the value as follows:

- A'(35), B'(35), C'(35), and D'(30)

Since avg (A', B', C', D') is equal to 33.75, the TeamSkill rule will be fulfilled. MatchTicket1 and MatchTicket2 can be successfully matched.

Team Assignment

Last updated : 2022-03-29 18:51:51

> Note :
> Since the product logic no longer meets the technical development of game industry, Game Player Matching GPM will be deprecated on June 1st, 2022. Please complete service migration before May 31 , 2022.

This document uses a rule as an example to describe how to slot players to team in GPM.

Sample Rule

The following rule shows a complete match of players to one red and two blue teams. Each team has 3 players in a 3v3v3 battle structure.

```
{
”version”: ”v1.0”,
”teams”: [{
”name”: ”red”,
”maxPlayers”: 3,
”minPlayers”: 3
}, {
”name”: ”blue”,
”maxPlayers”: 3,
”minPlayers”: 3,
”number”: [2,2]
}]
}
```

Use case 1: team is not specified

- **A single player initiates a match**

  Suppose the player A initiates a matchmaking request in which the `team` parameter is left empty, GPM may assign player A to red, blue 1, or blue 2 team. Then, the `team` parameter will be updated to "red_000", "blue_000", or "blue_001".

- **Multiple players initiate a team match**

Suppose players A and B initiate a team matchmaking request in which their `team` parameters are left empty, GPM will assign both players to the same team. Then, their `team` parameters will be both updated to "red_000", "blue_000", or "blue_001".

Use case 2: team is specified

- **A single player initiates a match**

  Suppose the player A initiates a matchmaking request in which the `team` parameter is specified to `red` , GPM will assign the player A to the red team. Then, the `team` parameter will be updated to "red_000".

- **Multiple players initiate a team match**

  Suppose both players A and B initiate a team matchmaking request in which their `team` parameters are specified to `blue` , GPM will assign both players to blue 1 or blue 2 team. Then, their `team` parameters will be both updated to "blue_000" or "blue_001".

> Note :
> Currently, players cannot specify different teams in a GPM team match.

Matchmaking Process

Battle Match

Last updated : 2022-03-29 18:51:51

Note :

Since the product logic no longer meets the technical development of game industry, Game Player Matching GPM will be deprecated on June 1st, 2022. Please complete service migration before May 31 , 2022.

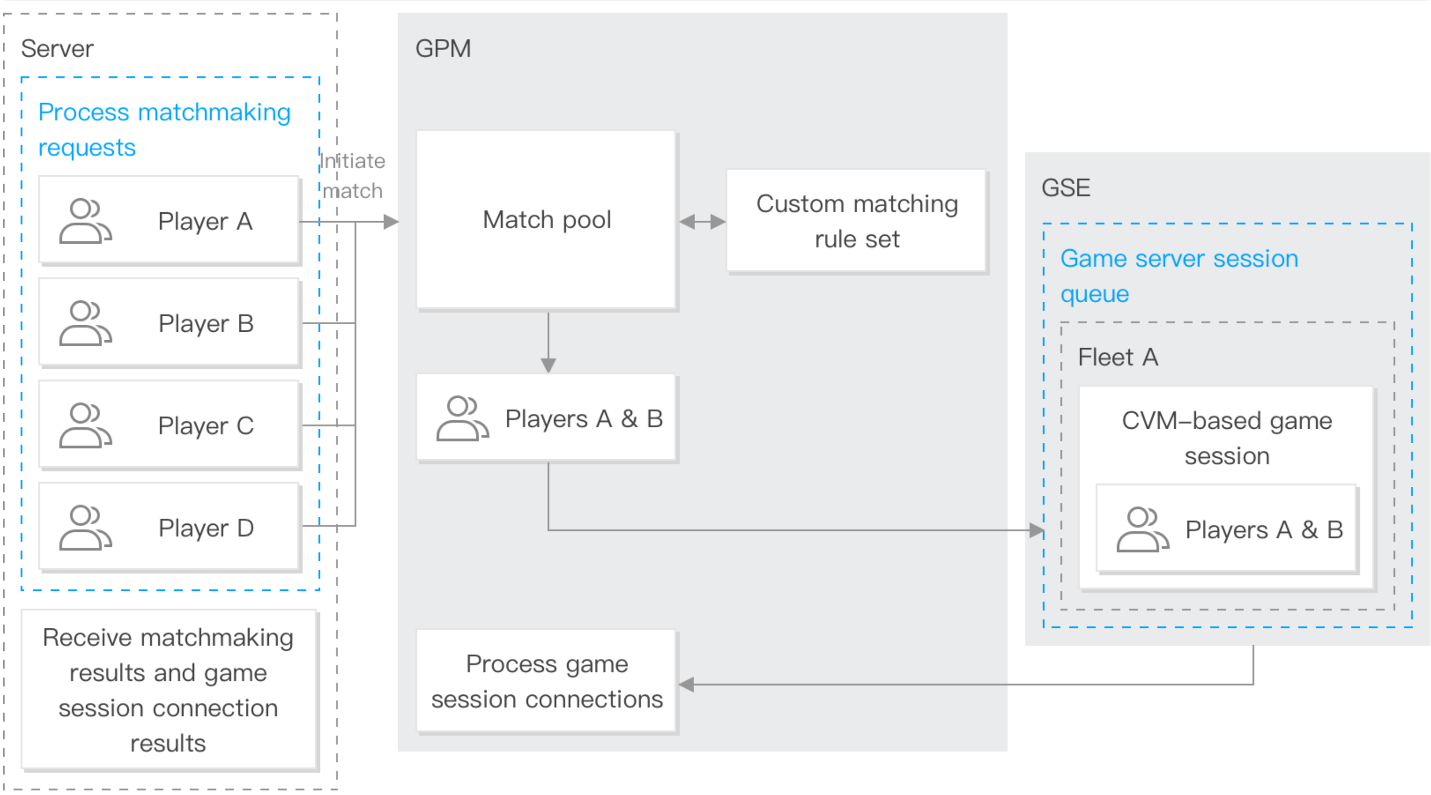
This document describes how to use GSE in combination with GPM to enhance your game development experience.

Prerequisites

- You have created a server fleet on the GSE console.
- You have created a match and selected **Request GSE Resources**.

Process Architecture

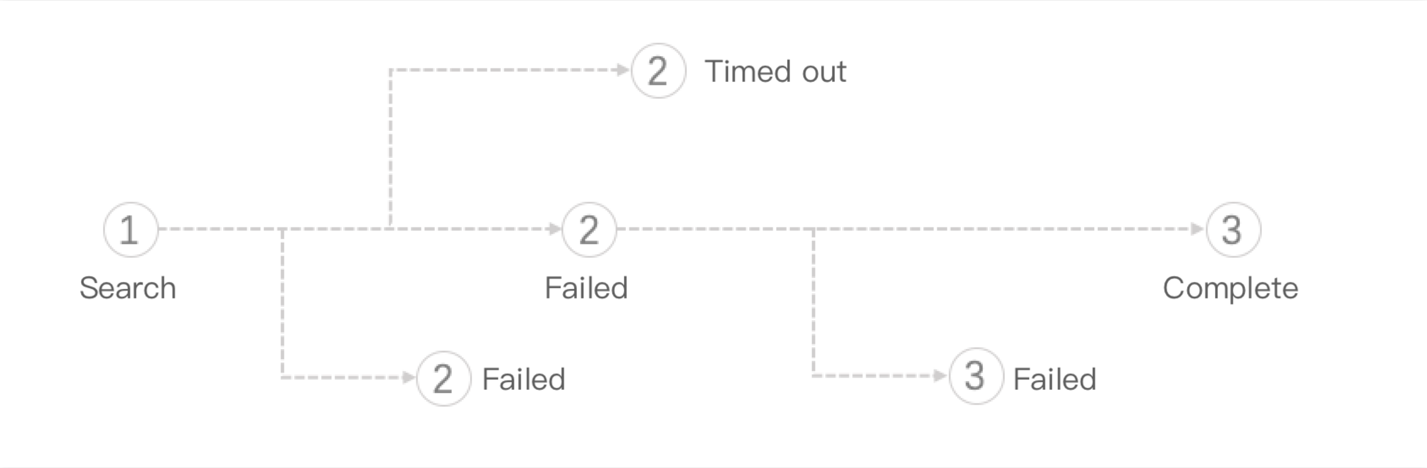
The process architecture of GSE+GPM is as follows:


Server
Process matchmaking requests
Player A
Player B
Player C
Player D
Initiate match
Receive matchmaking results and game session connection results
GPM
Match pool
Custom matching rule set
Players A & B
Process game session connections
GSE
Game server session queue
Fleet A
CVM–based game session
Players A & B

Matchmaking Status

To request GSE resources for the GPM matchmaking result, your MatchTicket will have the following statuses:

Note :
For more information about MatchTicket, see MatchTicket Parameters.



Directions

Step 1: initiate a matchmaking request on the client

You need a server to process matchmaking requests from your players, and call the `StartMatching` API to initiate matchmaking requests to GPM. A successful call of the `StartMatching` API generates a unique MatchTicket. You need to track the MatchTicket status to complete your matchmaking process.

You can call the `CancelMatching` API to cancel the matching MatchTicket.

Step 2: track the matchmaking status through event notifications

After a matchmaking request is initiated, you can configure a notification address to receive GPM event notifications. Meanwhile, you can set a timer to periodically process event notifications. For more information about event notification, see Event Notifications.

Step 3: process completed MatchTicket

- For a completed MatchTicket, GPM has already started a game server session on the game server queue, and filled in the relevant MatchTicket fields with game connections.
- You need to parse MatchTicket based on the protocol and obtain game connections as well as player ID and player session ID in the same battle.
- Players can access the active GSE game server session after receiving the game connections and player session ID. For more information about how to connect to a game server session, see Connecting Client to gRPC Server of GSE.
- For **cancelled/timeout/failed** ticket, you can notify players according to the actual business logic, or initiate a matchmaking request again for players.

(Recommended) Step 4: use the DescribeMatchingProgress API to complement event notifications

If you failed to configure a notification address or receive event notifications, you can call the `DescribeMatchingProgress` API to track the MatchTicket status. For more information about the statuses, see MatchTicket Status.

FAQs

1. How does the region latency of players affect GPM and GSE?

- **For GPM**
  - If there is no latency rule in the GPM rule script, the region latency of players will not be used.
  - If there is a latency rule in the GPM rule script, the region latency of all players matched to a battle should be satisfied in at least one region.

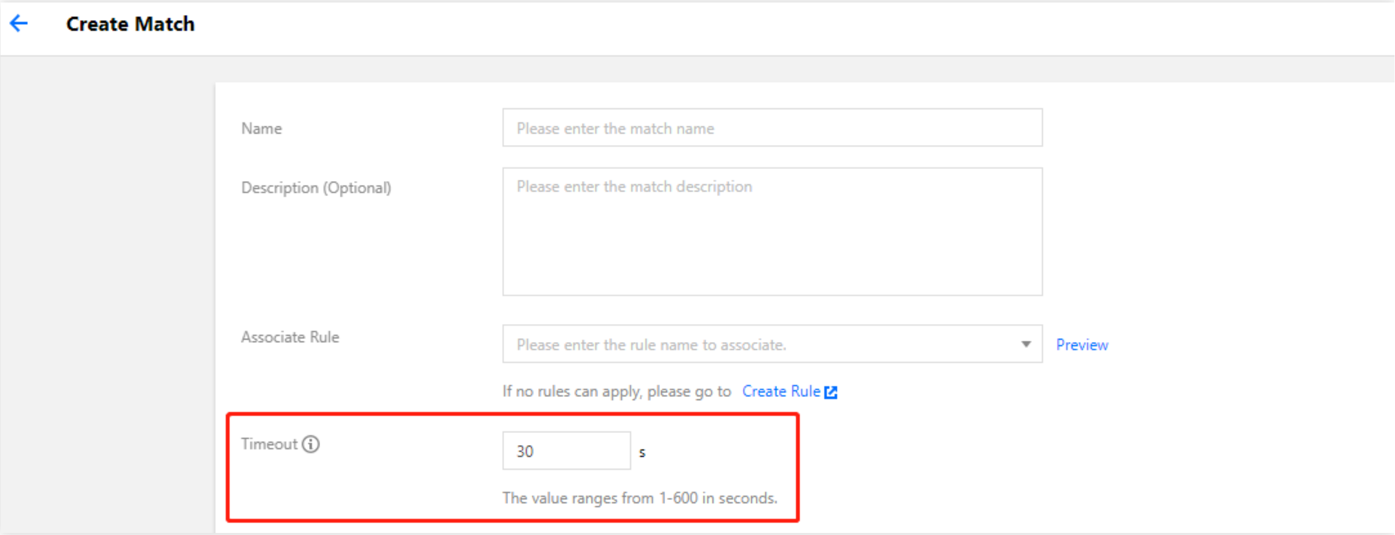
For more information about region latency in GPM, see Latency Rule.

- **For GSE**
  - If the region latency is not passed in when a matchmaking request is initiated, GSE will choose a fleet to place the game server session according to fleet priority in the queue.
  - If the region latency is passed in when a matchmaking request is initiated, GSE will use the region latency based on whether the queue is configured a latency policy.
    - No latency policy: GSE will place the game server session in the common region according to fleet priority.
    - Latency policy configured: GSE will place the game server session in the common region according to latency policy. If there are multiple eligible regions, GSE will choose the region according to fleet priority.

For more information about region latency in GSE, see Nearby Resource Scheduling.

2. What is the difference between timeouts in GPM and GSE?

- **GPM timeout**



Create Match

Name: Please enter the match name

Description (Optional): Please enter the match description

Associate Rule: Please enter the rule name to associate. Preview

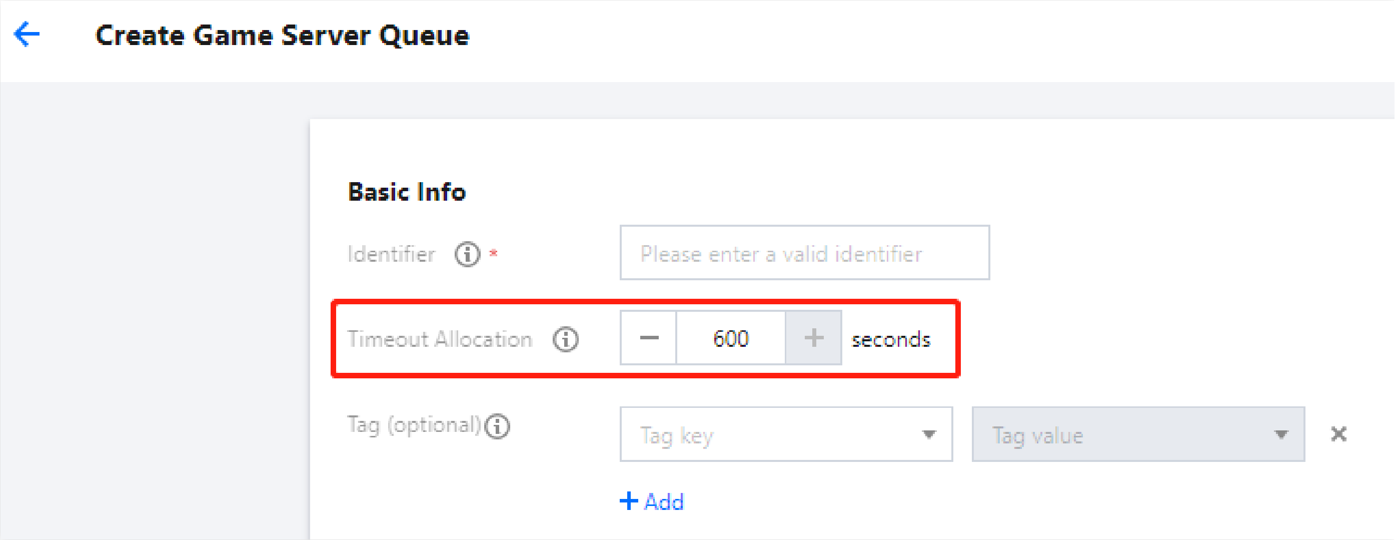
If no rules can apply, please go to Create Rule

Timeout: 30 s

The value ranges from 1-600 in seconds.

This timeout is the longest period during which GPM will search for a MatchTicket, and the MatchTicket is in `SEARCHING` status. If no other matched players are found during the period, the `MatchTicket` status will change to `TimedOut` .

- **GSE timeout**



Create Game Server Queue

Basic Info

Identifier *: Please enter a valid identifier

Timeout Allocation: 600 seconds

Tag (optional): Tag key / Tag value

This timeout is the longest period for retaining a game session request during which GSE will search for eligible fleet in the queue after a placement request is received. If no matched fleet is found during the period, the status of all `MatchTickets` involved in this placement change to `Failed` .

Note：

> The time consumed for a matchmaking service mainly includes the GPM searching and GSE placing.

Independent Match

Last updated : 2022-03-29 18:51:51

Note :
Since the product logic no longer meets the technical development of game industry, Game Player Matching GPM will be deprecated on June 1st, 2022. Please complete service migration before May 31 , 2022.

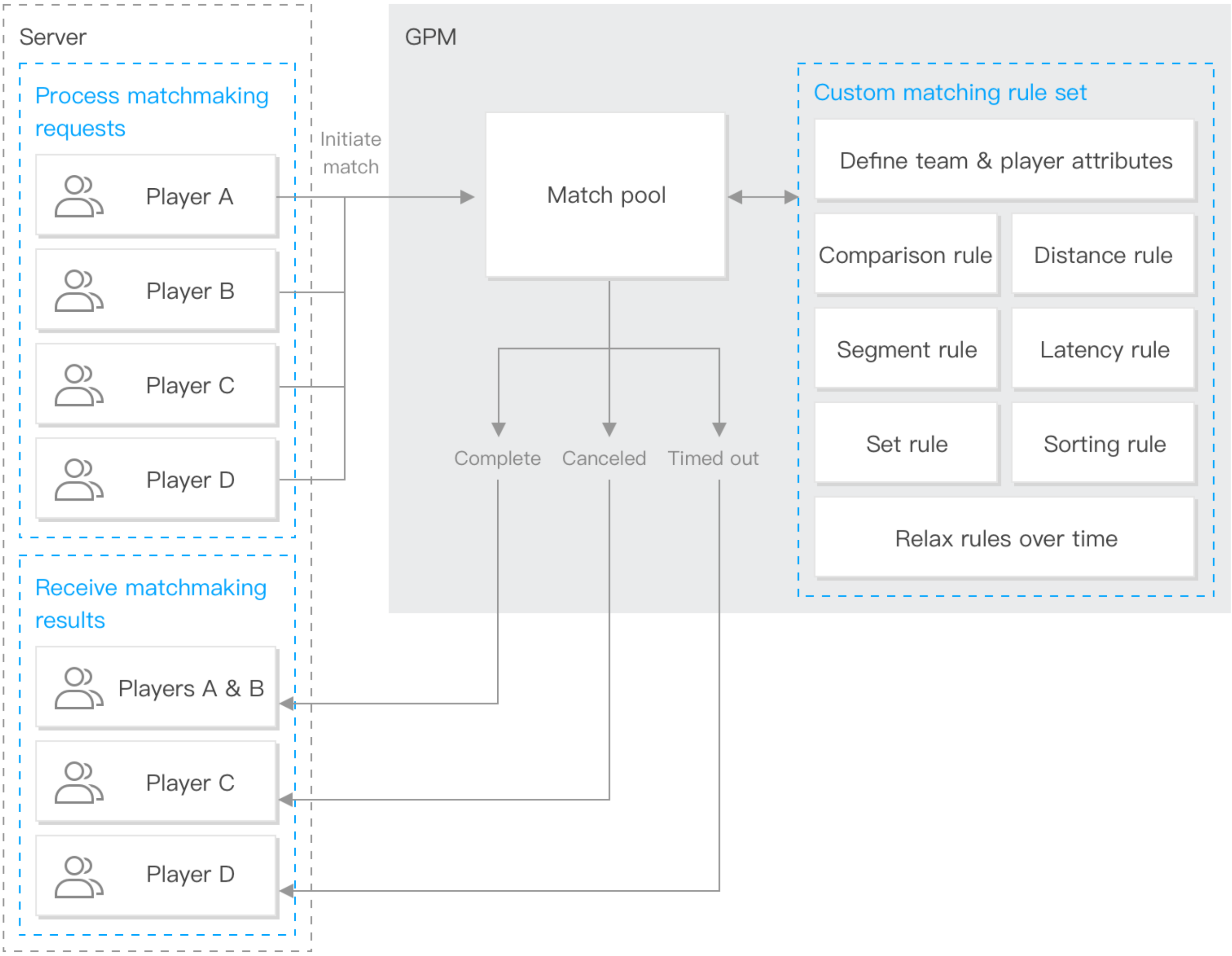
This document describes how to use GPM service alone.

Prerequisites

You have created a match without selecting **Request GSE Resources**.

Process Architecture

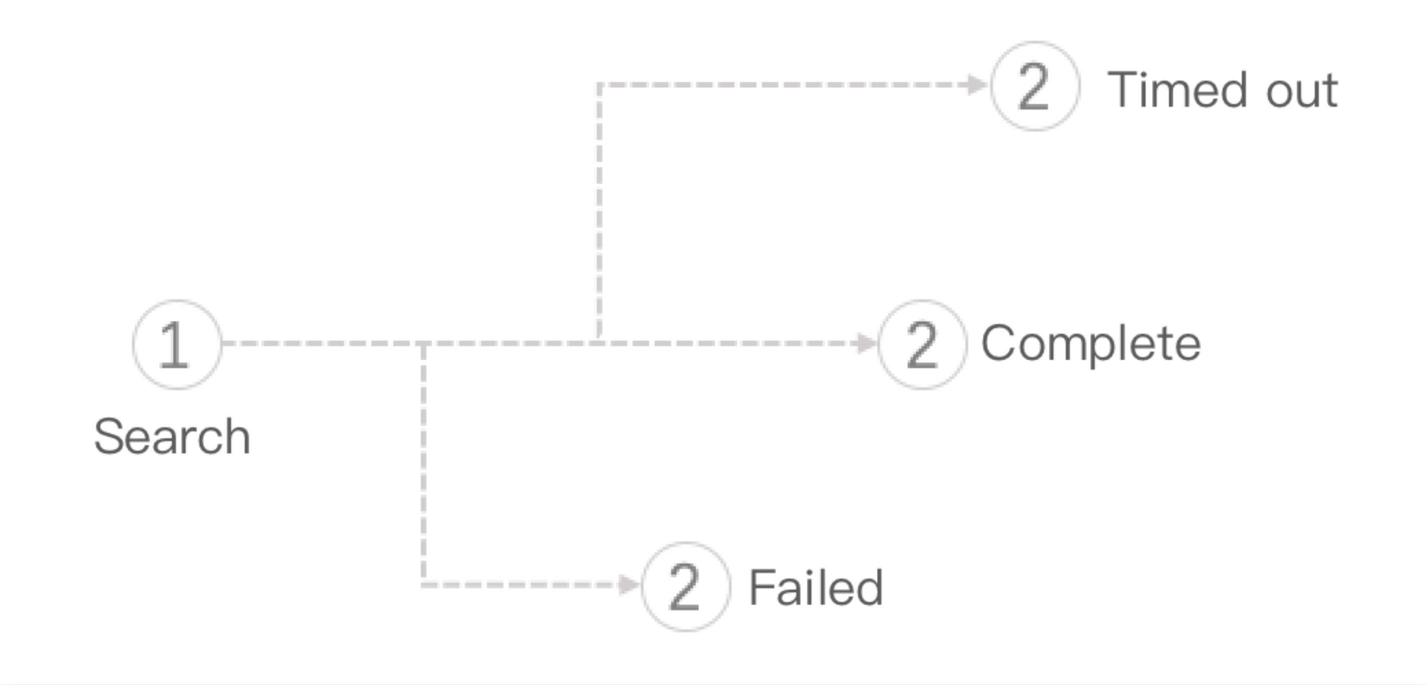
The GPM service can be independently used as follows:



Matchmaking Status

The MatchTicket statuses will be as follows when you use the GPM service alone.

> Note :
> For more information about MatchTicket, please see MatchTicket Parameters.



Directions

Step 1: initiate a matchmaking request on the client

You need a server to process matchmaking requests from your players, and call the `StartMatching` API to initiate matchmaking requests to GPM. A successful call of the StartMatching API generates a unique MatchTicket. You need to track the MatchTicket status to complete your matchmaking process.

You can call the `CancelMatching` API to cancel the matching MatchTicket.

Step 2: track the matchmaking status through event notifications

After a matchmaking request is initiated, you can configure a notification address to receive GPM event notifications. Meanwhile, you can set a timer to periodically process event notifications. For more information about event notification, see Event Notifications.

Step 3: process completed MatchTicket

- After the potential battle ticket is found, you need to parse MatchTicket based on the protocol and obtain the player IDs matched to the same battle.
- For eligible battle tickets, you need a server to process matchmaking requests from players and connect these players to the same battle. Then, you need to process the player connection and battle logic in GPM.

- For **cancelled/timeout/failed** ticket, you can notify players according to the actual business logic, or initiate a matchmaking request again for players.

(Optional) Step 4: use the DescribeMatchingProgress API to complement event notifications

If you failed to configure a notification address or receive event notifications, you can call the `DescribeMatchingProgress` API to track the MatchTicket status. For more information about the statuses, see MatchTicket Status.

Matchmaking Rule

Rule Script Design Guide

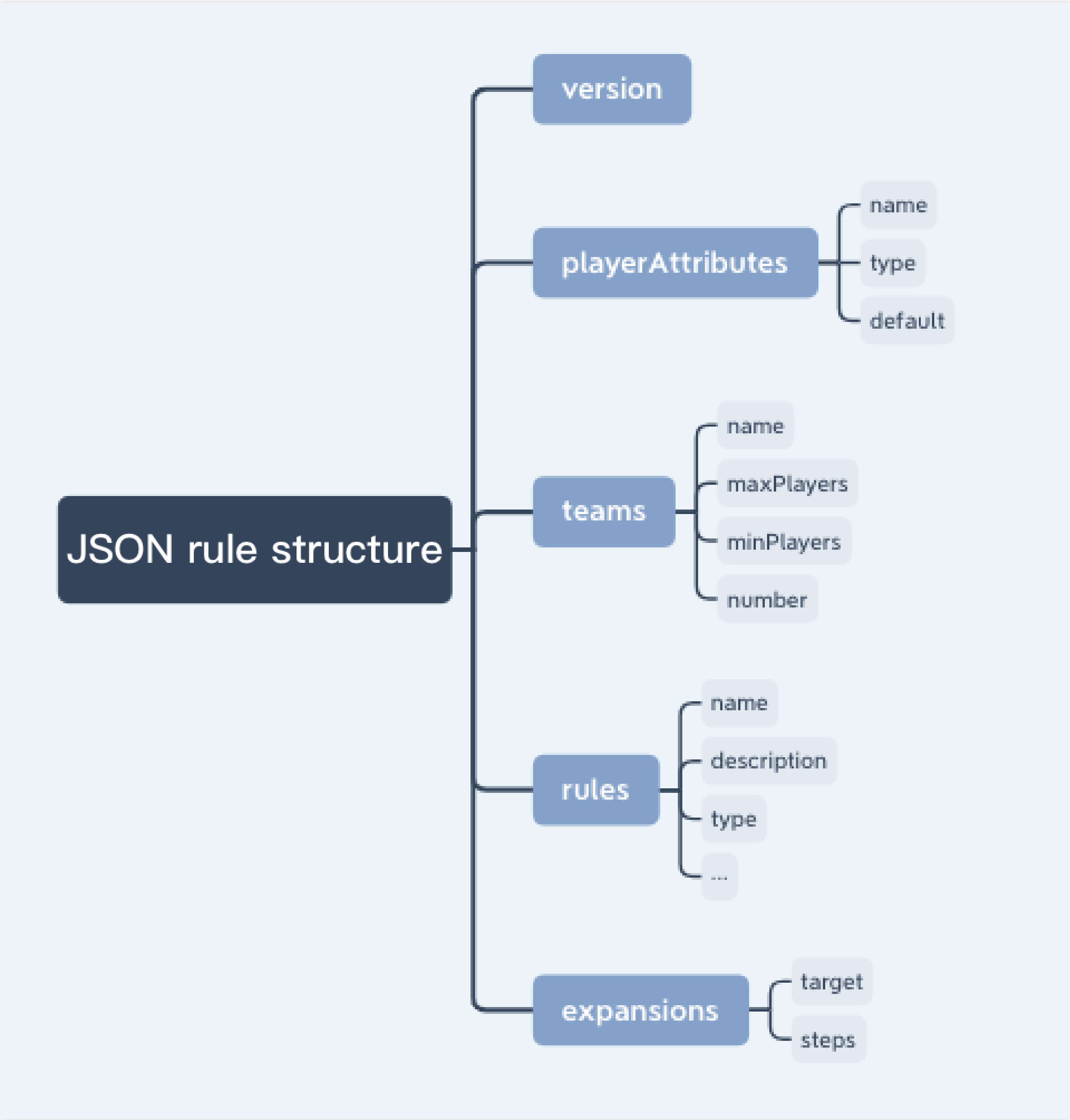
Last updated : 2022-03-29 18:51:51

> Note :
> Since the product logic no longer meets the technical development of game industry, Game Player Matching GPM will be deprecated on June 1st, 2022. Please complete service migration before May 31 , 2022.

This document describes how to design a script for matching rules.

JSON Rule Structure

The structure of a matchmaking rule script is as follows:



Parameter Details

version

This required parameter describes the version information. Currently, its value can only be "v1.0", as shown below:

```
"version": "v1.0"
```

playerAttributes

This optional parameter is an array consisting of 0-5 player attributes.

- **Parameters**

Parameter	Type	Required	Description
name	string	Yes	Attribute name, which corresponds to the attribute name passed in to the StartMatching API. The `name` parameter in the array is unique, which contains letters and digits only.
type	string	Yes	Attribute type. Valid values: `number`, `string`
default	number/string	No	Default value of the attribute specified in `type`. ◦ If the attribute value is not passed in a matchmaking request, the default value will be used. ◦ If no default value is provided, a matchmaking request must specify the attribute value.

- **Sample code**

```
"playerAttributes": [
{
"name": "skill",
"type": "number",
"default": 10
},
{
"name": "gameMode",
"type": "string",
"default": "classic"
}
]
```

teams

This required parameter indicates all team types in a complete match. Currently, a team consists of up to 40 players.

- **Parameters**

Parameter	Type	Required	Description
name	string	Yes	Team type. The `name` parameter in the array is unique, which contains letters and digits.
maxPlayers	number	Yes	Maximum number of players in a team of this type
minPlayers	number	Yes	Minimum number of players in a team of this type
number	number_list	No	◦ Number of teams of this type. ◦ The `number_list` parameter only contains two elements in the format of [MinTeamCount, MaxTeamCount]. ◦ If this parameter is not specified, the default value [1,1] will be used, indicating that a complete match involves only one team of this type . ◦ Currently, `MinTeamCount` must be equal to `MaxTeamCount, and a team must have fixed number of players.

- **Sample code**

```
"teams": [
{
"name": "soldier",
"maxPlayers": 4,
"minPlayers": 4,
"number": [2,2]
},
{
"name": "king",
"maxPlayers": 4,
"minPlayers": 4
}
]
```

rules

This optional parameter is an array consisting of 0-10 matchmaking rules. Valid values: `distanceRule` , `comparisonRule` , and `latencyRule` . For more information about rules, see Rule Types.

- **Parameters**

Parameter	Type	Required	Applicable Rule
name	string	Yes	All rules
type	string	Yes	All rules
description	string	No	All rules
measurements	-	-	distanceRule, comparisonRule
referenceValue	-	-	distanceRule, comparisonRule
operation	-	-	comparisonRule
maxDistance	-	-	distanceRule, latencyRule
minDistance	-	-	distanceRule
maxLatency	-	-	latencyRule
distanceReference	-	-	latencyRule
partyAggregation	string	No	All rules

> Note :
> The “Type” and “Required” fields of the `measurements` , `referenceValue` , `operation` , `maxDistance` , `minDistance` , `maxLatency` and `distanceReference` parameters are determined by the rule type. For more information, see Rule Types.

- **Sample code**

```
"rules": [{
"name": "FairTeamSkill",
"type": "distanceRule",
"measurements": [ "avg(teams[*].players.playerAttributes[skill])" ],
"referenceValue": "avg(flatten(teams[*].players.playerAttributes[skill]))",
"maxDistance": 10
}, {
"name": "RedTeamSelection",
"type": "comparisonRule",
”operation”: ”=”,
”measurements”: [”flatten(teams[red].players.playerAttributes[colour])”],
”referenceValue”: ”red”
}, {
”name”: ”FastConnection”,
”type”: ”latencyRule”,
”maxLatency”: 50
}]
```

expansions

A matchmaking rule expansion contains 0-5 targets that consist of 1-10 steps. You can configure `waitTimeSeconds` to adjust the limitations of the matchmaking rule.

- **Parameters**

Parameter		Type	Required	Description	Valid Value
target		string	Yes	Target to be adjusted	◦ Teams: minPlayers, maxPlayers ◦ distanceRule: minDistance, maxDistance ◦ latencyRule: maxLatency, maxDistance
steps	waitTimeSeconds	number	Yes	Wait time in second	number
	value	number	Yes	Updated target value after `waitTimeSeconds`	number

- **Sample code**

```
”expansions”: [{
”target”: ”teams[soldier].minPlayers”,
”steps”: [{
”waitTimeSeconds”: 5,
”value”: 6
}, {
”waitTimeSeconds”: 15,
”value”: 4
}]
}, {
”target”: ”rules[FastConnection].maxLatency”,
”steps”: [{
”waitTimeSeconds”: 10,
”value”: 100
}, {
”waitTimeSeconds”: 20,
”value”: 150
}]
}]
```

Expressions

Operator

Operator	Input	Output	Description
avg	List< number > List< List< number >>	number, List< number >	Obtains the average value of the list.
flatten	List< List< ? >>	List< ? >	Changes the two-dimensional list to a flat list that contains all elements.

Expression

Expression	Description	Result Type	Usage
teams[soldier].maxPlayers	Maximum number of players in the \`soldier\` team	number	• Used to configure \`target\` under \`expansions\`

Expression	Description	Result Type	Usage
teams[soldier].players.playerAttributes[xxx]	List of attributes xxx of players in the \`soldier\` team	List< List< ? >>	- Used to configure \`measurements\` under \`rules\` - Used to configure \`referenceValue\` under \`rules\`
teams[*].players.playerAttributes[xxx]	List of attributes xxx of players in all teams	List< List< ? >>	- Used to configure \`measurements\` under \`rules\` - Used to configure \`referenceValue\` under \`rules\`
teams[soldier].players[playerId]	List of IDs of players in the \`soldier\` team	List< List< string >>	- Used to configure \`measurements\` under the rule type \`comparisonRule\`
rules[FairTeamSkill].maxDistance	The \`maxDistance\` parameter of the \`FairTeamSkill\` rule	number	- Used to configure \`target\` under \`expansions\`

Rule Types

Last updated : 2022-03-29 18:51:51

Note :
Since the product logic no longer meets the technical development of game industry, Game Player Matching GPM will be deprecated on June 1st, 2022. Please complete service migration before May 31 , 2022.

This document describes GPM matching rules.

Difference Rule

A difference rule is used to limit the difference between the measurements and the reference value.

- **Attributes**

Attribute	Description	Required	Valid Value
name	Rule name	Yes	The rule name in one `RuleScript` is only contain letters and digits.
type	Rule type	Yes	`distanceRule`
description	Rule description	No	Strings
measurements	Measured value. It is used to compare with the `referenceValue`	Yes	An operation expression that can be number or list of numbers. For exam ◦ "avg(teams[*].players.playerAttril ◦ "avg(flatten((teams[*].players.pla
referenceValue	Comparison reference value	Yes	A number or an operation expressio parsed into a numeric value. For exa ◦ 60 ◦ "avg(flatten((teams[*].players.pla
minDistance	Minimum distance	`minDistance` and `maxDistance`	A number between 0 to 99999.

		cannot be both empty.	
maxDistance	Maximum distance	\`minDistance\` and \`maxDistance\` cannot be both empty.	A number between 0 to 99999
partyAggregation	Processing method for multiplayer MatchTicket	No	○ Valid values: \`each\`, \`min\`, \`max ○ Default value: \`each\` ○ For more information about \`party Multi-player Match

- **Implementation**

  GPM will search for suitable MatchTicket that allows the absolute value of difference between `measurements` and `referenceValue` to be no more than `maxDistance` and no less than `minDistance` .

  - If the `measurement` expression is parsed into a number, it will be compared with `referenceValue` .
  - If the `measurement` expression is parsed into a list of numbers, each of its elements will be compared with `referenceValue` .

- **Samples**

  The following sample codes show that the average skill deviation of each team from that of all teams in a game battle is no more than 10.

```
”rules”: [{
”name”: ”FairTeamSkill”,
”type”: ”distanceRule”,
”measurements”: [ ”avg(teams[*].players.playerAttributes[skill])” ],
”referenceValue”: ”avg(flatten(teams[*].players.playerAttributes[skill]))”,
”maxDistance”: 10
}]
```

Comparison Rule

A comparison rule is used to indicate the relationship between string or numeric attributes by comparison operators.

- **Attributes**

Attribute	Description	Required	Valid Value
name	Rule name	Yes	The rule name in one `RuleScript` is uniqu only contain letters and digits.
type	Rule type	Yes	`comparisonRule`
description	Rule description	No	Strings
measurements	Measured value. It is used to compare with `referenceValue`	Yes	An operation expression that can be parse number, list of numbers, string, or list of s example, ◦ "flatten(teams[*].players.playerAttribut ◦ "avg(flatten(teams[*].players.playerAttr
referenceValue	Comparison reference value	No	A number, string, or an operation expressi be parsed into a number, such as "avg(flatten((teams[*].players.playerAttrib
operation	Comparison operator	Yes	Valid values: `=`, `!=`, `<`, `<=`, `>`, `> strings, the valid values include `=` and `
partyAggregation	Processing method for multiplayer MatchTicket	No	◦ Valid values: `each`, `min`, `max`, `avg ◦ Default value: each ◦ For more information about `partyAggre Multi-player Match

- **Implementation**

GPM will search for suitable MatchTicket that allows the relationship between `measurements` and `referenceValue` to satisfy the comparison operator specified in `operation` .

- If the `measurement` expression is parsed into a number, it will be compared with `referenceValue` .
- If the `measurement` expression is parsed into a list of numbers, each of its elements will be compared with `referenceValue` .
- If the `measurement` expression is parsed into a string, it will be judged according to whether it is the same as `referenceValue` .
- If the `measurement` expression is parsed into a list of strings, each of its elements will be judged according to whether each is the same as `referenceValue` .
- If `referenceValue` is not specified, each element of `measurements` will take the defined player attribute or not according to `operation` .

- **Samples**

    - Sample 1: players matched should choose the same gameMode

    ```
    "rules": [{
    "name": "SameGameMode",
    "type": "comparisonRule",
    "operation": "=",
    "measurements": ["flatten(teams[*].players.playerAttributes[gameMode])"]
    }]
    ```

    - Sample 2: the color attribute of the red team players must be `red`

    ```
    "rules": [{
    "name": "RedTeamSelection",
    "type": "comparisonRule",
    "operation": "=",
    "measurements": ["flatten(teams[red].players.playerAttributes[colour])"],
    "referenceValue": "red"
    }]
    ```

Latency Rule

A latency rule is used to match players to servers according to the region latency, ensuring matched players from different regions have similar response time.

> Note :
> GPM matches players according to the region latency without updating or choosing regions for players. After players are successfully matched based on latency rules, at least one region meets the rule conditions for battle players. Then, GSE will choose the region for battle servers. For more information, see Nearby Resource Scheduling.

- **Attributes**

Attribute	Description	Required	Valid Value
name	Rule name	Yes	The rule name in one `RuleScript` is unique, which can only

			contain letters and digits.
type	Rule type	Yes	latencyRule
description	Rule description	No	Strings
maxLatency	Acceptable maximum latency to the service region	Yes	A number between 0 to 999999
maxDistance	Maximum difference between the latency of each player to the service region and `distanceReference`	No. It must be used with `distanceReference`.	A number between 0 to 999999
distanceReference	Reference value	No. It must be used with `maxDistance`.	Valid values: `min`, `avg` ◦ `min`: the minimum latency of all matched players to a region ◦ `avg`: the average latency of all matched players to a region
partyAggregation	Processing method for multi-player MatchTicket	No	◦ Valid values: `each`, `min`, `max`, `avg`, `any` ◦ Default value: `each` ◦ For more information about `partyAggregation`, see Multi-player Team Match

- **Implementation**

Each player inputs the latency from their respective location to every region when sending a matchmaking request. GPM will search for suitable MatchTicket that allows at least one region to meet the latency rule for matched players.

> Note :
> GPM does not support latency test. You need to do so by yourself or use the Latency Test Tool provided by GSE.

- **Samples**

  - Sample 1: all players matched to a game have latencies of no more than 150 ms to at least one region

    ```
    ”rules”: [{
    ”name”: ”lowLatency”,
    ”type”: ”latencyRule”,
    ”maxLatency”: 150
    }]
    ```

  - Sample 2: all players matched to a game have latencies of no more than 150 ms to at least one region, and their latency difference is less than 80 ms.

    ```
    ”rules”: [{
    ”name”: ”lowLatency”,
    ”type”: ”latencyRule”,
    ”maxLatency”: 150,
    ”maxDistance”: 80,
    ”distanceReference”:”min”
    }]
    ```

Rule Examples

Last updated : 2022-03-29 18:51:51

> Note :
> Since the product logic no longer meets the technical development of game industry, Game Player Matching GPM will be deprecated on June 1st, 2022. Please complete service migration before May 31 , 2022.

This document describes typical rule examples.

4V4V4 Difference and Latency Rules

Use case: the average skill difference between each team and all matched teams is no more than 10 when a matchmaking request is initiated. In addition, all players have latencies of no more than 50 ms to at least one region. Add an expansion rule, which will be relaxed a while later.

```
{
”version”: ”v1.0”,
”playerAttributes”: [
{
”name”: ”skill”,
”type”: ”number”,
”default”: 10
}
],
”teams”: [
{
”name”: ”red”,
”maxPlayers”: 4,
”minPlayers”: 4
},
{
”name”: ”green”,
”maxPlayers”: 4,
”minPlayers”: 4
},
{
”name”: ”blue”,
”maxPlayers”: 4,
”minPlayers”: 4
}
],
”rules”: [
{
”name”: ”FairTeamSkill”,
”type”: ”distanceRule”,
”measurements”: [
”avg(teams[*].players.playerAttributes[skill])”
],
”referenceValue”: ”avg(flatten(teams[*].players.playerAttributes[skill]))”,
”maxDistance”: 10
},
{
”name”: ”FastConnection”,
”type”: ”latencyRule”,
”maxLatency”: 50
}
],
”expansions”: [
{
”target”: ”rules[FairTeamSkill].maxDistance”,
”steps”: [
{
”waitTimeSeconds”: 5,
”value”: 50
},
{
”waitTimeSeconds”: 15,
”value”: 100
}
]
},
{
”target”: ”rules[FastConnection].maxLatency”,
”steps”: [
{
”waitTimeSeconds”: 10,
”value”: 100
},
{
”waitTimeSeconds”: 20,
”value”: 150
}
]
}
]
}
```

XV8 Asymmetric Matching and Comparison Rule

Use case: two teams with different number of players are matched to a game battle, where team A has variable players, and team B has fixed players. These players should select the same game mode.

```
{
”version”: ”v1.0”,
”playerAttributes”: [{
”name”: ”gameMode”,
”type”: ”string”,
”default”: ”turn-based”
}],
”teams”: [{
”name”: ”red”,
”maxPlayers”: 7,
”minPlayers”: 2
}, {
”name”: ”blue”,
”maxPlayers”: 8,
”minPlayers”: 8
}],
”rules”: [ {
”name”: ”SameGameMode”,
”type”: ”comparisonRule”,
”operation”: ”=”,
”measurements”: [”flatten(teams[*].players.playerAttributes[gameMode])”]
}]
}
```

0V5 Man-machine Battle

Use case: a team of 5 players are matched to a battle against a user-defined robot

```
{
”version”: ”v1.0”,
”playerAttributes”: [{
”name”: ”skill”,
”type”: ”number”
}],
”teams”: [{
”name”: ”Marauders”,
”maxPlayers”: 5,
”minPlayers”: 5
}],
”rules”: [{
”name”: ”lowLatency”,
”description”: ”Sets maximum acceptable latency”,
”type”: ”latencyRule”,
”maxLatency”: 150
}],
”expansions”: [{
”target”: ”rules[lowLatency].maxLatency”,
”steps”: [{
”waitTimeSeconds”: 12,
”value”: 200
}]
}]
}
```

2V2 One of the Two Teams without a Constraint Rule

```
{
”version”: ”v1.0”,
”playerAttributes”: [{
”name”: ”skill”,
”type”: ”number”
}],
”teams”: [{
”name”: ”red”,
”maxPlayers”: 2,
”minPlayers”: 2
},{
”name”: ”blue”,
”maxPlayers”: 2,
”minPlayers”: 2
}],
”rules”: [{
”name”: ”redTeamRule”,
”type”: ”distanceRule”,
”measurements”:[”teams[red].players.playerAttributes[skill]”],
”referenceValue”:10,
”maxDistance”: 5
}]
}
```

Event Notifications

Notification Mechanism

Last updated : 2022-03-29 18:51:51

> Note :
> Since the product logic no longer meets the technical development of game industry, Game Player Matching GPM will be deprecated on June 1st, 2022. Please complete service migration before May 31 , 2022.

This document describes GPM event notifications to help you track the status of your matchmaking requests.

Prerequisites

- You have created a service address to receive event notifications.
- You have created a match and configured the URL service address to receive event messages.

Event Notification Protocol

Field	Type/Value	Description
Time	string	Timestamp of an event notification, such as "1970-01-01T00:00:00Z"
Account	string	Tencent Cloud account UIN
Region	string	Region of the match
Source	string	Valid value: `TencentCloud.GPM`
Type	string	Type of event messages
Id	string	Unique ID of the event, which is generated by GPM
MatchTickets	Array of MatchTicket	List of MatchTickets of an event. The MatchTicket is either passed in by developers or generated by GPM when each matchmaking request is initiated.

Field	Type/Value	Description
CustomEventPushData	string	Custom push data, which is defined when a match is created.
MatchTokens	Array of string	List of effective MatchTokens, which is used to verify the message reliability. For more information about MatchToken usage, see Event Notification Security.

Event Type

- **MatchmakingSearching**
  The MatchTicket of the matchmaking request is being searched by GPM.
- **MatchCreated**
  The MatchTicket of the matchmaking request can be matched to a battle that meets the rule conditions.
- **MatchmakingSucceeded**
  The MatchTicket of the matchmaking request is placed in the specified GSE server queue, for which, GSE makes game server session and player session ready for the upcoming battle, and waits for player connections.
- **MatchmakingTimedOut**
  The MatchTicket of the matchmaking request timed out, and no eligible players are found.
- **MatchmakingCancelled**
  The MatchTicket of the matchmaking request has been cancelled.
- **MatchmakingFailed**
  The MatchTicket of the matchmaking request failed.

Event Notification Timing

Depending on the event type, different messages will be pushed to the URL you configured.

- The **MatchmakingSearching** event will be triggered when GPM starts searching for a MatchTicket.
- The **MatchmakingCancelled** event will be triggered when a MatchTicket is cancelled. In this case, its lifecycle in GPM ends.
- The **MatchmakingTimedOut** event will be triggered when a MatchTicket is not matched to a potential battle within the specified timeout period. In this case, its lifecycle in GPM ends.

- The **MatchmakingFailed** event will be triggered when an error occurs to a MatchTicket at any stage. In this case, its lifecycle in GPM ends.
- The **MatchCreated** event will be triggered after a MatchTicket **matched** to a potential battle.
- For a match without requesting GSE resources, all MatchTickets of the **match** finish their GPM lifecycles.
- For a match requesting GSE resources, the statuses of all MatchTickets of the **match** change to **PLACING**.
- The **MatchmakingSucceeded** event will be triggered when a **match** requesting GSE resources is successfully placed to a GSE game server queue. In this cause, all MatchTickets of the **match** finish their GPM lifecycle.

Event Notification Security

Last updated : 2022-03-29 18:51:51

Note :
Since the product logic no longer meets the technical development of game industry, Game Player Matching GPM will be deprecated on June 1st, 2022. Please complete service migration before May 31 , 2022.

This document describes the security mechanism of GPM event notifications to help you identify reliable event messages and improve business security.

Prerequisites

- You have created a match and obtained a MatchCode.
- When creating or editing a match, you have configured the URL notification address for the MatchCode.

MatchToken Introduction

MatchToken acts as a key that is used to verify the reliability of MatchCode event notifications. This verification can increase your business security and avoid forged messages.

MatchToken Generation and Update

You need to call the `ModifyToken` API to manually generate or update a **MatchToken** for a match. When specifying request parameters of this API, you can specify `MatchToken` parameter to customize a token. If this parameter is left empty, a random token will be generated by GPM. You also need to specify `CompatibleSpan` , during which, GPM will send both the old and new MatchTokens.

For a MatchCode, the MatchToken change interval should be at least three minutes.

MatchToken Query

Call the `DescribeToken` API to query the latest MatchToken of the specified MatchCode.

MatchToken Verification

You need to record the latest MatchToken after every setting or update. Check whether this MatchToken is the same as that of the event message.

For a MatchCode configured with MatchToken, its **MatchTokens** field of each event message contains all effective MatchTokens (including the latest and all MatchTokens within the specified `CompatibleSpan` period).