CONSTRUCT GAME SERVICES

View online: https://www.construct.net/en/game-services/manuals/game-services

Construct Game Services

We offer a range of services to help you build more interactive and better functioning games.

CREATE A GAME

View online: https://www.construct.net/en/game-services/manuals/game-services/games/create-game

Create a Game

A game in Construct Services should be thought of as a project. Each game has it's own set of players and it's own resources such as leaderboards.

Player bases cannot be shared between games.

To create a game, visit your Construct Service Account page. Once a game is created, you can create services that belong to this game.

Test & Production

We recommend that for testing you create a separate game to run your test code from, and create another game for production.

API KEYS

View online: https://www.construct.net/en/game-services/manuals/game-services/games/api-keys

Game API Keys

Some resources or requests you make on a game may require an API key.

API keys should NEVER be exposed or distributed to clients.

Create API Key

In your Construct Services Account, go to `Your Games`, select the game you wish to create an API key for then select `API Keys` from the left menu.

When you create an API key, the key will be shown to you in full once only. Make sure you record this safely and secretly. Once you continue to the next page, it will not be possible to show you the full key again.

Suspend Key

When viewing your games API keys, you can suspend a key which will immediately stop the key from working. You can unsuspend suspended keys at any time.

Delete Key

Deleting a key permanently removes the key from your game and prevents it from being usable again. All API requests coming through using this key will start failing.

If you wish to implement a new API key, we would recommend creating a new key, suspending the old key and once you confirm no more requests are incoming for the old key you can safely delete it.

CONCEPTS

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/concepts

What is the Authentication Service?

When you create a game, you will need to register players or allow players to sign in the game before allowing them to interact with other services.

Once a player is registered or signed in, a session key will be returned for this player. Passing this session key into other services allows this player to interact with other Construct Game Services.

All players have a unique player ID. Players also have a **player name** which should be used to display publicly. If a player signs in with a username and password it should be noted that the **username** is different to the **player name** and is only used to authenticate the sign in.

Registering Players

There are two ways to register players:

- Specific call to register player
- Signing in a player with a login provider

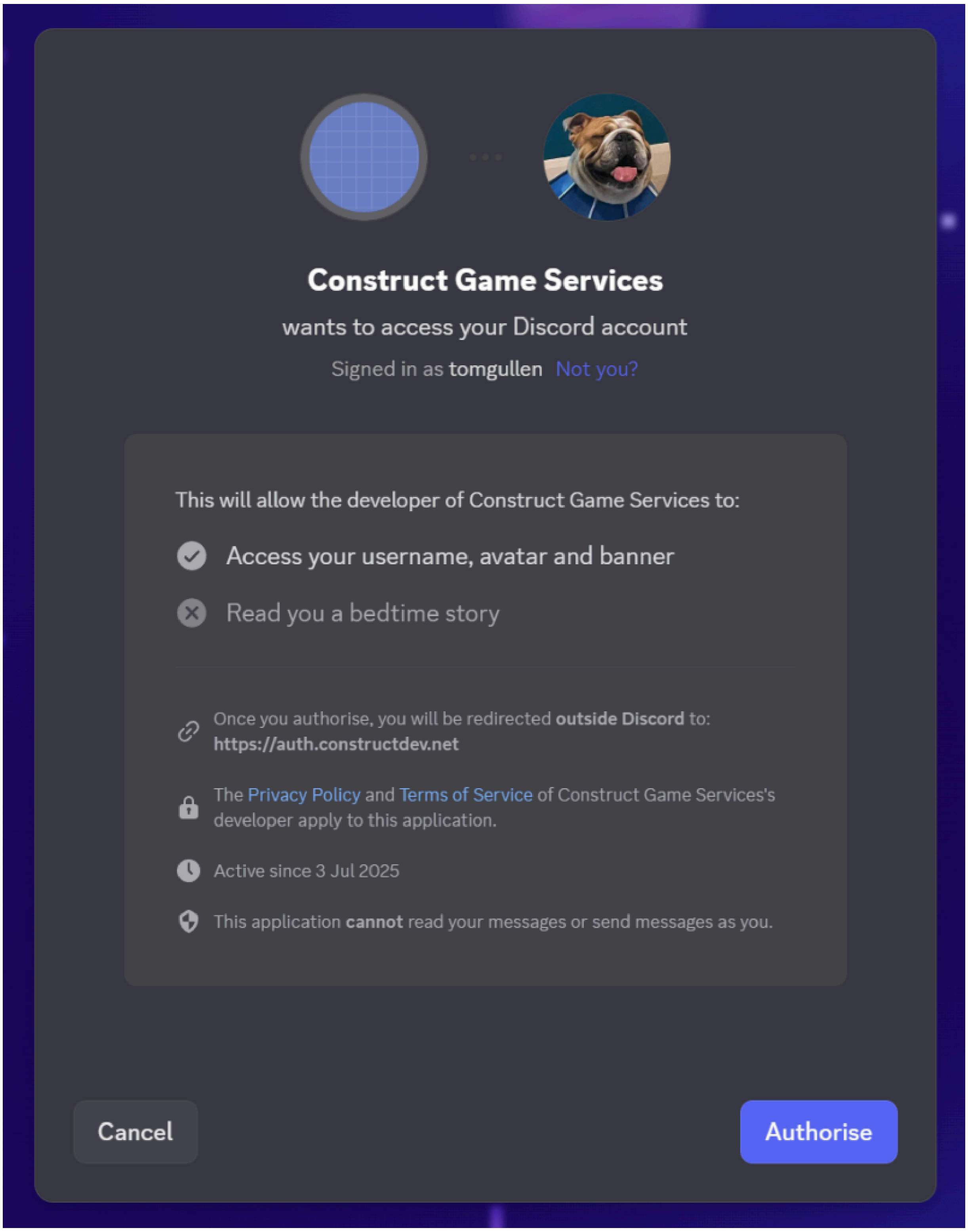
If you register a player manually and **don't** provider a username and password, if the session key expires or is lost this player account will be unrecoverable. It is advised if possible to encourage the player to set a username/password, or link to a login provider at some point to ensure they can use this newly created account.

When a player signs in with a login provider, a new player will be registered if this player has not previously signed into your game. If the player has previously signed in with this login provider, a new session key will be generated which ties them to their already established player ID.

SUPPORTED LOGIN PROVIDERS

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/login-providers

We offer support for many different login providers for signing in players. Below is an example of what players see when signing in with Discord into your game:


Construct Game Services

wants to access your Discord account

Signed in as **tomgullen** Not you?

This will allow the developer of Construct Game Services to:

✓ Access your username, avatar and banner

✗ Read you a bedtime story

Once you authorise, you will be redirected **outside Discord** to: **https://auth.constructdev.net**

The Privacy Policy and Terms of Service of Construct Game Services's developer apply to this application.

Active since 3 Jul 2025

This application **cannot** read your messages or send messages as you.

In your CGS account, you can configure settings for each login provider allowing more customisation as to what the players see when signing in.

Difficulty and expertise requried for each login provider varies a lot. Some simple require simple setting changes, others such as Apple require far more complex integration.

Permission Minimalisation

For all login providers, the bare minimum permissions will be requested from the login provider to provide sign in capability for CGS. We do not request permissions from players accounts that we do not need. The purpose of login providers for CGS is purely to support a login mechanism.

Signing In

When signing in with a login provider, sometimes the provider will return an avatar. If the player does not yet have an avatar, the login providers avatar will be set as this players current avatar.

Linking Login Providers

Players can have multiple login providers allowing them to sign into the same player account. To link another login provider to an existing player account, use the link end point.

When linking a login provider to an existing account, sometimes the login provider being linked is already associated with another player account. In such circumstances, a **force token** will be generated allowing you to call the force link end point. Force linking a login provier will remove it from the existing players account, and assign it to the requesting players account. It's important to note that you should always confirm with the player they wish to force this link, as if this was the only login provider on the other player account it will become unrecoverable.

Supported Providers

Username/Password

Allows players to sign in with a username and password. Passwords are stored on CGS servers hashed with BCrypt.
Usernames must be unique within the game, you cannot have two players with the same login username. Usernames must be between 3 and 32 characters in length.
Passwords must be between 8 and 32 characters in length, contain at least 1 number, 1 special character, 1 upper case letter and 1 lower case letter.

Apple

Allows players to sign in with their Apple account.

Facebook

Allows players to sign in with their Facebook account.

Google

Allows players to sign in with their Google account.

Microsoft

Allows players to sign in with their Microsoft account.

Steam

Allows players to sign in with their Steam account.

Discord

Allows players to sign in with their Discord account.

BattleNet

Allows players to sign in with their BattleNet account.

BattleNet China

Allows players to sign in with their Chinese BattleNet account. Blizzard separate player bases by regions which is why BattleNet China is treated as a separate login provider.

X (Twitter)

Allows players to sign in with their X (Twitter) account.

Reddit

Allows players to sign in with their Reddit account.

Yandex

Allows players to sign in with their Yandex account.

PLAYER SESSIONS

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/sessions

Sessions

When a player registers or signs in, a session key will be generated. Each session has a default expiry of 24 hours. The API end points you use allow an optional parameters to specify an explicit expiry if you want shorter/longer expiries.

A player can only have one active session. If the player signs in from another device, a new session key will be generated and the old session key will no longer be usable.

Session keys should be stored on the client so they can be re-used.

Refreshing Sessions

Session expiries can be extended by calling the refresh session end point. If you use short expiries, it is important to keep refreshing the session to make sure the player is no unexpectedly signed out when interacting with your game.

Ending a Session

If you wish to sign out a player, you can simply remove the session key from the client device so that it is no longer known or can be used. However, when signing out we recommend for security purposes to also make a call to the end session end point. Calling this invalidates the session key itself.

SIGN IN FLOW

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/sign-in-flow

Sign In Flow

Signing in a player requires a few steps to complete successfully.

Call signin.json

Call the sign in API end point. For username/password sign in this will return if the sign in was successful or not immediately.

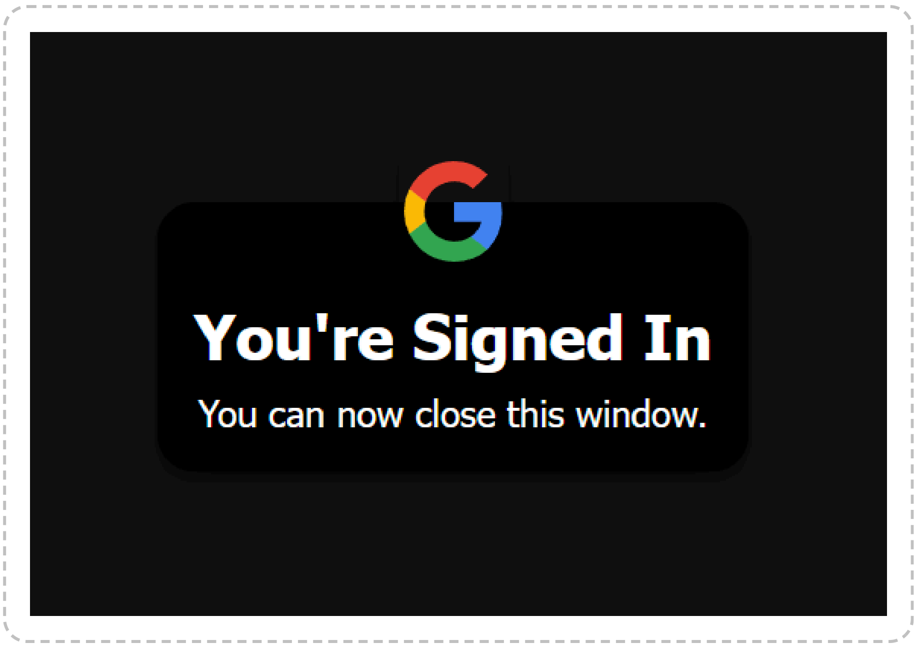
For other login providers, a redirect URL will be returned along with a sign in poll token:

```
{
  "redirectToURL": "https://accounts.google.com/o/oauth2/v2/auth....",
  "pollToken": "e1d03c89-d03d-459f-ad2d-e7607be375e0",
  "success": true,
  "debug_requestTimeMS": 0
}
```

Redirect User

The redirect URL should preferably be opened on the client in a new popup window. This opens the third party login providers sign in page.

Once the sign in succeeds, the client is redirected to a successful sign in page that looks similar to the following:


You're Signed In

You can now close this window.

At this point the user can close the window and through polling with the poll token you can then shortly afterwards determine if the user succeeded in the sign in or not. It should be noted that this page has some Javascript code on it:

```
    <script>
        window.opener.postMessage("LOGINCOMPLETED", "*");
        window.close();
    </script>
```

If the client allows it, a message is sent to the window opened indicating the login is completed, then attempt to automatically close.

Determine If Sign In Was Successful

With the poll token, and query the sign in poll API end point periodically. This will tell you when the sign in fails or succeeds.

We recommend querying this end point every 500-1000ms. If you query the poll token too frequently you may receive rate limit errors. If you query the same poll token before previous poll requests have completed you will always see an error. You should only re-query the poll token when the previous poll token response has completed and returned data.

THE PLAYER OBJECT

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-objects/player-object

The Player Object

When you retrieve a player, a player object will be returned.

Example Player Object

```
{
  "id": "8478281d-88dd-429e-9a96-d08a3f37631c",
  "playerName": "Tom",
  "created": "2025-01-28T11:40:46.2",
  "avatars": [...]
}
```

Object Properties

id guid

A unique record ID for this player.

playerName string

The publicly facing player name for this player.

created datetime

The date this player was first registered in the game.

avatars

If player has an avatar, a list of avatar objects. Each avatar object is the same avatar, but provided in different sizes. Sizes available are based on widths, and the widths 16, 32, 64, 96, 128 and 256 will be shown here along with the original avatar size if it is different to this predetermined list. Some widths may not show if the original avatar width is smaller than any of the available widths.

EXPANDED PLAYER

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-objects/expanded-player

The Expanded Player Object

The get players API end point requires a game secret key to be passed, which returns expanded player objects which contain more information about a player that should not be public.

Example Expanded Player Object

```
{
  "id": "8478281d-88dd-429e-9a96-d08a3f37631c",
  "playerName": "Tom",
  "created": "2025-01-28T11:40:46.2",
  "successfulSignIns": 45,
  "lastSuccessfulSignIn": "2025-02-15T11:40:46.2",
  "loginProviders": [ ... ],
  "avatars": [ ... ]
}
```

Object Properties

id guid

A unique record ID for this player.

playerName string

The publicly facing player name for this player.

created datetime

The date this player was first registered in the game.

successfulSignIns int

Total number of times this player has signed into their account.

lastSuccessfulSignIn datetime

Date and time of the last successful sign in for this player.

loginProviders

List of player login provider objects currently assigned to this player.

avatars

If player has an avatar, a list of avatar objects. Each avatar object is the same avatar, but provided in different sizes. Sizes available are based on widths, and the widths 16, 32, 64, 96, 128 and 256 will be shown here along with the original avatar size if it is different to this predetermined list. Some widths may not show if the original avatar width is smaller than any of the available widths.

THE SESSION OBJECT

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-objects/session-object

The Session Object

When you retrieve a session, a session object will be returned.

Example Session Object

```
{
  "key": "abc...",
  "playerID": "8478281d-88dd-429e-9a96-d08a3f37631c",
  "playerName": "Tom",
  "gameID": "9584f985-b378-431c-a869-28be0a95172b",
  "expiry": "2025-01-28T11:40:46.2",
  "avatars": [...]
}
```

Object Properties

key string

The session key for this player.

playerID guid

The player ID this session belongs to.

playerName string

The publicly facing player name for this player.

gameID guid

The game ID this player belongs to.

expiry datetime

The expiry date time of this session. Can be extended by calling refresh session API end point.

avatars

If player has an avatar, a list of avatar objects. Each avatar object is the same avatar, but provided in different sizes. Sizes available are based on widths, and the widths 16, 32, 64, 96, 128 and 256 will be shown here along with the original avatar size if it is different to this predetermined list. Some widths may not show if the original avatar width is smaller than any of the available widths.

PLAYER LOGIN PROVIDER OBJECT

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-objects/player-login-provider

The Player Login Provider Object

When calling the API end point to retrieve all available player login providers, this object will be returned for each login provider currently attached to the player.

Example Player Login Provider Object

```
{
  "playerID": "8478281d-88dd-429e-9a96-d08a3f37631c",
  "provider": "Discord",
  "providerID": 2,
  "avatarURL": "https://avatars.discord.com.../avatar.png",
  "signIns": 6,
  "firstSignIn": "2025-01-20T11:40:46.2",
  "lastSignIn": "2025-01-28T12:39:41.5",
  "username": null
}
```

Object Properties

playerID guid

The player ID this login provider belongs to.

provider string

The provider. Will be one of: UsernamePassword, Facebook, Discord, X, Reddit, Yandex, Google, Steam, Apple, BattleNet or BattleNetChina.

providerID int

A unique ID for the provider.

avatarURL string

The URL to the avatar of this user within this login provider if it exists. This image should probably not be used anywhere publicly, but allows for you to present the player with the option to set their player avatar to their Discord avatar if they wish to. If the player changes their avatar with this login provider, the next time they sign in with this login provider the value will be updated.

signIns int

The number of successful sign ins this player has had with this player account using this login provider.

firstSignIn datetime

The date and time of their first sign in with this login provider.

lastSignIn datetime

The date and time of their last successful sign in with this login provider.

username string

If the login provider is **UsernamePassword** this players username will be returned in the response.

THE AVATAR OBJECT

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-objects/avatar

The Player Object

When you retrieve a player, a player object will be returned.

Example Avatar Object

```
{
  "width": 64,
  "height": 64,
  "url": "https://auth.construct.net/.../file.png"
}
```

Object Properties

width int

The width in pixels of the avatar.

height iny

The height in pixels of the avatar.

url string

The URL this avatar is located at.

PAGINATION OBJECT

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-objects/pagination

The Pagination Object

When a response contains or could contain multiple records, a pagination object is returned.

Example Pagination Object

```
{
    "requestedPage": 5,
    "totalPages": 1003,
    "recordsPerPage": 2,
    "totalRecords": 2005,
    "prevPage": 4,
    "nextPage": 6
}
```

Object Properties

requestedPage int32

The page of results returned in this result set.

totalPages int32

The total pages of results returned in this result set.

recordsPerPage int32

How many records per page are being returned in this result set.

totalRecords int32

Total number of records in this result set.

prevPage int32

The previous page number. Will not be returned in the response if this is the first page in this result set.

nextPage int32

The next page number. Will not be returned in the response if this is the last page in this result set.

CHANGE PLAYER PASSWORD

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/players/change-password

Change a Players Password

If a player has a username + password set, you can change their password with this end point. You can see if a player has a username & password set by querying the get all login providers end point.

If a player does not have a username + password set, you should call the link end point to create a username + password for this player.

Passwords must be between 8 and 32 chars in length, contain at least 1 number, 1 special character (non alpha numeric), 1 upper case character and 1 lowercase character.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/changepassword.json
```

Authenticating The Request

There are two ways to authenticate this request. One with a **secret** and **playerID**, the other with a **sessionKey**.

Secret Key Authentication

secret string

Your games secret key.

playerID guid

The player ID you wish to make this request against.

password string

The current password for this player.

Session Key Authentication

sessionKey string

The session key of the player you're making the request against.

Request Parameters

gameID guid Required

The game ID you're making the request against.

newPassword string Required

The new password for the player.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Password must contain at least 1 upper case character.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

CHANGE PLAYER NAME

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/players/change-player-name

Change a Players Player Name

A players name is the publicly facing name that will be shown to other players, for example on leaderboard scores. Player names cannot exceed 50 characters in length. Player names are unique for each game, you cannot have 2 or more players with the same player name.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/changeplayername.json
```

Authenticating The Request

There are two ways to authenticate this request. One with a **secret** and **playerID**, the other with a **sessionKey**.

Secret Key Authentication

secret string

Your games secret key.

playerID guid

The player ID you wish to make this request against.

Session Key Authentication

sessionKey string

The session key of the player you're making the request against.

Request Parameters

---

gameID guid Required

The game ID you're making the request against.

---

playerName string Required

The new player name.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Player name is already in use!",
  "shouldRetry":false
}
```

---

success bool

If the request was successful or not. For request failures, this will always be false.

---

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

---

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```json
{
  "success": true
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

DELETE A PLAYER

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/players/delete-player

Delete a Player

Deleting a player removes them from your game completely, and deletes all associated data from all other Construct Game Services in your game such as leaderboard scores, cloud saves etc. Deletion is completely irreversible.

It's important to ensure that you give players the option to delete themselves for privacy reasons.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/deleteplayer.json
```

Authenticating The Request

There are two ways to authenticate this request. One with a **secret** and **playerID**, the other with a **sessionKey**.

Secret Key Authentication

secret string

Your games secret key.

playerID guid

The player ID you wish to make this request against.

Session Key Authentication

sessionKey string

The session key of the player you're making the request against.

Request Parameters

gameID guid Required

The game ID you're making the request against.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Session invalid.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true
}
```

---

success bool

If the request was successful or not. For request successes, this will always be true.

GET PLAYER

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/players/get-player

Get Player

This end point allows you to retrieve a player object based on the players player name. As the only way to query this end point is with a secret key, this should only be called by back end services.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/getplayer.json
```

Authenticating The Request

secret string

Your games secret key.

Request Parameters

gameID guid Required

The game ID you're making the request against.

playerName string

The player name you are querying. Either playerName or playerID must be specified.

playerID guid

The player ID you are querying. Either playerName or playerID must be specified.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Player name does not exist in game.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true,
  "player": { ... }
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

player player

The player object for the found player.

GET PLAYERS

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/players/get-players

Get Players

This end point allows you to retrieve expanded player objects. As the only way to query this end point is with a secret key, this should only be called by back end services.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/getplayers.json
```

Authenticating The Request

secret string

Your games secret key.

Request Parameters

gameID guid Required

The game ID you're making the request against.

If requesting players by their known ID's, also provide the following parameters:

playerIDs string Required

A CSV list of player ID's.

If you're not querying know player ID's and instead want a list of players, use the following parameters:

page int

Page of results you're fetching

perPage int

Maximum number of players to return on each page of results. Default is 50, and the maximum allowed is 200.

order string

One of AZ, ZA, Newest or Oldest. Defaults to AZ, which returns players ordered by their player name alphabetically. Newest/oldest return players based on the date they were created.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Player name does not exist in game.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-

attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true,
  "players": [ ... ],
  "pagination": { ... }
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

players

A list of expanded player objects for your request.

pagination

If not requesting known player IDs pagination object will be included giving you information about other results.

REGISTER A PLAYER

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/players/register-player

Register a New Player

Creates a new player in your game.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/registerplayer.json
```

Authenticating The Request

This request requires no authentication unless your game is configured to require a secret for all authentication post requests.

Request Parameters

gameID guid Required

The game ID you're making the request against.

playerName string Required

The player name you are attempting to register.

username string

If you want this player to be able to login with a username and password in the future, specify their username here.

password string

If you want this player to be able to login with a username and password in the future, specify their password here.

---

expiryMins int

Registering a new player returns a session key for this newly created player if succcessful. Optional provider the session key expiry in minutes for this newly created session. Must be at least 5, and no more than 129600 (90 days). Default session expiry is 24 hours.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Player name already taken.",
  "shouldRetry":false
}
```

---

success bool

If the request was successful or not. For request failures, this will always be false.

---

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

---

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true,
  "player": { ... },
  "session": { ... },
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

player player

The player object for the newly created player.

session session

The session object for the newly created session for this player.

SET USERNAME + PASSWORD

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/players/set-username-password

This end point allows you to set a username + password for a player to sign in with. If the player already has a username + password set, this will overwrite their existing details.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/setusernamepassword.json
```

Authenticating The Request

There are two ways to authenticate this request. One with a **secret** and **playerID**, the other with a **sessionKey**.

Secret Key Authentication

secret string

Your games secret key.

playerID guid

The player ID you wish to make this request against.

password string

The current password for this player.

Session Key Authentication

sessionKey string

The session key of the player you're making the request against.

Request Parameters

username string Required

The requested username to login with.

password string Required

The requested password to login with.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Player name does not exist in game.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true,
  "players": [ ... ],
  "pagination": { ... }
}
```

---

success bool

If the request was successful or not. For request successes, this will always be true.

---

players

A list of expanded player objects for your request.

---

pagination

If not requesting known player IDs pagination object will be included giving you information about other results.

SIGN IN

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/players/sign-in

Sign In

The first step of signing a player in with a specified login provider.

If a sign in succeeds but the login provider is not tied to any player account a new player account will be created.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/signin.json
```

Authenticating The Request

No authentication is required for this request.

Request Parameters

gameID guid Required

The game ID you're making the request against.

provider string Required

The login provider you want to sign the player in with. Must be one of UsernamePassword, Facebook, Discord, X, Reddit, Yandex, Google, Microsoft, Steam, Apple, BattleNet or BattleNetChina.

expiryMins int

Registering a new player returns a session key for this newly created player if succcessful. Optional provider the session key expiry in minutes for this newly

created session. Must be at least 5, and no more than 129600 (90 days).
Default session expiry is 24 hours.

username string

If provider is set to UsernamePassword, this username parameter must be specified.

password string

If provider is set to UsernamePassword, this password parameter must be specified.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage": "Invalid login provider specified.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

If signing in with UsernamePassword login provider, you will know immediately if the sign in was successful or not:

```
{
  "success": true,
  "session": { ... }
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

session session

The new session object of the successful sign in.

If signing in with a third party login provider, the response will be different as this is part of a multi step sign in flow.

```
{
  "success": true,
  "redirectToURL": "https://auth.battlenet.com/....",
  "pollToken": "4109a99a-7a34-4e87-8331-90673921c20a"
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

redirectToURL string

Open this URL in a new window for the client to continue with their login.

pollToken guid

A token to query the poll sign in API end point to determine if the sign in succeeds or not.

SIGN IN POLL

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/players/sign-in-poll

Sign In Poll

When making a sign in request, some sign ins will return a URL to visit as part of the sign in flow. Typically this redirects the user to a third party login provider. Along with this URL a poll token will be provided.

You can use this poll token to query the sign in to monitor if it succeeds or fails. Once it succeeds, a session object is returned.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/signinpoll.json
```

Authenticating The Request

pollToken guid

The sign in poll token provided on the initial sign in request

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Poll token is invalid.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

---

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

---

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code. Until the sign in resolves (the user cancels the sign in flow or the sign in succeeds) the response will only include the **success** parameter.

If sign in fails, (for example the user cancels the sign in) **signInFailed** will be true, and an error message under the **signInErrorMessage** property will be returned.

If sign in succeeds, **session** will return the new session data.

In both circumstances, the returned data will only be available once. Repeated calls with the same poll token will fail unless the sign in is still in progress.

It's not always possible to determine if a sign in has been abandoned. In such cases **success** will keep returning true until the poll token expires (in approximately 5 minutes from the initial sign in request).

```
{
  "success": true,
  "signInFailed": true,
  "signInErrorMessage": "User cancelled sign in",
  "session": { ... }
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

signInFailed bool

If true, indicates the sign in failed or was abandoned.

signInErrorMessage string

Details about why the sign in failed if it failed or was abandoned.

session session

The new session object if the sign in was succesful.

END SESSION

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/sessions/end-session

End a Player Session

Ends a player session. Calling this end point invalidates the session key, requiring the player to sign in again in the future to create a new session.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/endsession.json
```

Authenticating The Request

sessionKey string

The session key of the player you're making the request against.

Request Parameters

gameID guid Required

The game ID you're making this request against.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Invalid session key.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

GET PLAYERS SESSION

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/sessions/get-session

Get a Player Session

Retrieves a players session.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/getsession.json
```

Authenticating The Request

sessionKey string

The session key of the player you're making the request against.

Request Parameters

gameID guid Required

The game ID you're making this request against.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Invalid session key.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true,
  "session": { ... }
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

session session

The session object returned in your request.

REFRESH SESSION

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/sessions/refresh-session

Refresh a Player Session

Extends the life of a players session by the initial set session expiry when signing in the player.

If on sign in, the players session expiry was set to 6 hours, upon refreshing the session the new session expiry will now be 6 hours in the future.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/refreshsession.json
```

Authenticating The Request

sessionKey string

The session key of the player you're making the request against.

Request Parameters

gameID guid Required

The game ID you're making this request against.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Invalid session key.",
  "shouldRetry":false
}
```

---

success bool

If the request was successful or not. For request failures, this will always be false.

---

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

---

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true
}
```

---

success bool

If the request was successful or not. For request successes, this will always be true.

DISCONNECT LOGIN PROVIDER

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/login-providers/disconnect

Disconnect Login Provider

Disconnect a login provider from a users account. They will no longer be able to login with this login provider unless they link it again.

If all login providers for a players account are disconnected, the account may become unrecoverable so it is advisable to ensure they always have one login provider available.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/disconnect.json
```

Authenticating The Request

A session key must be provided to authenticate the request.

sessionKey string

The session key of the player you're making the request against.

Request Parameters

provider string Required

The provider to disconnect. Must be one of UsernamePassword, Facebook, Discord, X, Reddit, Yandex, Google, Microsoft, Steam, Apple, BattleNet or BattlenetChina.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Invalid session key.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

FORCE LINK LOGIN PROVIDER

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/login-providers/force-link

Force Link Login Provider

When linking a login provider to a user, if the login provider is associated with another player account you can call this end point to force the login provider to be linked to this player account, disconnecting it from the other account.

You should not call this automatically, but present the player the option to force the link or cancel. If you force the link, it can make the other account unrecoverable.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/forcelink.json
```

Authenticating The Request

A session key must be provided to authenticate the request.

code guid

The force link code provided from the initial link request.

Request Parameters

There are no other request parameters for this request.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Invalid session key.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

GET ALL LOGIN PROVIDERS

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/login-providers/get-login-providers

Get All Players Login Providers

Returns all the login providers currently associated with a player.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/getconnectedloginproviders.json
```

Authenticating The Request

A session key must be provided to authenticate the request.

sessionKey string

The session key of the player you're making the request against.

Request Parameters

There are no additional request parameters for this request.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Invalid session key.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true,
  "connectedProviders": [ ... ]
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

connectedProviders

A list of login provider objects for this player.

LINK LOGIN PROVIDER

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/login-providers/link-login-provider

Link Login Provider

If a player is signed in, you can link another login provider to this player.

If provider is set to UsernamePassword, you must provide a proposed username and password in your request. This will immediately return if the username/password successfully set for this player or not.

For other login providers, a redirect URL will be returned along with a poll token to complete the link request in the same way as a sign in request.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/link.json
```

Authenticating The Request

A session key must be provided to authenticate the request.

sessionKey string

The session key of the player you're making the request against.

Request Parameters

gameID guid Required

The game ID you're making the request against.

provider string Required

The login provider you want to sign the player in with. Must be one of UsernamePassword, Facebook, Discord, X, Reddit, Yandex, Google, Microsoft, Steam, Apple, BattleNet or BattleNetChina.

---

username string

If provider is set to UsernamePassword, this username parameter must be specified.

---

password string

If provider is set to UsernamePassword, this password parameter must be specified.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage": "Invalid login provider specified.",
  "shouldRetry":false
}
```

---

success bool

If the request was successful or not. For request failures, this will always be false.

---

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

---

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

If linking with UsernamePassword login provider, you will know immediately if the sign in was successful or not:

```
{
  "success": true
}
```

---

success bool

If the request was successful or not. For request successes, this will always be true.

---

session session

The new session object of the successful sign in.

If linking with a third party login provider, the response will be different:.

```
{
  "success": true,
  "redirectToURL": "https://auth.battlenet.com/....",
  "pollToken": "4109a99a-7a34-4e87-8331-90673921c20a"
}
```

---

success bool

If the request was successful or not. For request successes, this will always be true.

---

redirectToURL string

Open this URL in a new window for the client to continue with their link.

---

pollToken guid

A token to query the poll sign in API end point to determine if the link succeeds or not.

You can query the sign in poll API end point to determine if the link was successful or not. This will only return **success: true** once - subsequent requests with the same poll token will fail.

Unlike signing in, no new session will be created with a link request.

DELETE PLAYER AVATAR

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/avatars/delete-avatar

Delete Player Avatar

Deletes a players avatar.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/deleteavatar.json
```

Authenticating The Request

There are two ways to authenticate this request. One with a **secret** and **playerID**, the other with a **sessionKey**.

Secret Key Authentication

secret string

Your games secret key.

playerID guid

The player ID you wish to make this request against.

Session Key Authentication

sessionKey string

The session key of the player you're making the request against.

Request Parameters

gameID guid Required

The game ID you are making the request against.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Invalid session key.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

SET PLAYER AVATAR

View online: https://www.construct.net/en/game-services/manuals/game-services/authentication/api-end-points/avatars/set-avatar

Delete Player Avatar

Sets a players avatar, overwriting the existing one if the player already has an avatar set. Supported avatar file types are PNG, SVG, WEBP, JPG or GIF.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://auth.construct.net/setavatar.json
```

Authenticating The Request

There are two ways to authenticate this request. One with a **secret** and **playerID**, the other with a **sessionKey**.

Secret Key Authentication

secret string

Your games secret key.

playerID guid

The player ID you wish to make this request against.

Session Key Authentication

sessionKey string

The session key of the player you're making the request against.

Request Parameters

gameID guid Required

The game ID you are making the request against.

avatar string

The base64 encoded avatar image. Either **avatar** or **avatarURL** must be sent with this request.

avatarURL string

The URL to the avatar image. Either **avatar** or **avatarURL** must be sent with this request.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Invalid image format.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

GETTING STARTED WITH LEADERBOARDS

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/getting-started

Create a Leaderboard

To create a leaderboard, visit your Construct Services account page. Create a game if you have not already, and from there you can add leaderboards.

Construct 3 Plugin

It's easy to interact with the leaderboard service directly within Construct 3 using the official Construct Services Plugin.

Construct Arcade

If you use the Construct Services Plugin in your Construct 3 project, the plugin has methods to interact with the Construct Arcade Leaderboard. When you upload your game to the Construct Arcade a leaderboard will be created and configured automatically for you. Please note, for scores to be submitted to the Construct Arcade users must be logged in on Construct.net, guest scores are not permitted.

.NET Class Library

If you use .NET, you can install the Construct Services class library to use on your website or application.

The Construct.net website itself uses this library to run the Construct Arcade leaderboards, so will be well maintained and new features added as and when they are released.

Installation

Using the .NET Core command-line interface (CLI) tools:

```
dotnet add package ConstructServices
```

Using the NuGet Command Line Interface (CLI):

```
nuget install ConstructServices
```

Using the Package Manager Console:

```
Install-Package ConstructServices
```

CONSTRUCT LEADERBOARD CAPABILITIES

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/capabilities

Construct Leaderboard Capabilities

The leaderboard service is designed to be scalable and high performance allowing you to rely on it for your leaderboard needs!

Although this service is branded towards games, the leaderboard service is suitable for ranking anything - competitions, sales teams, events, sports or anything else you can think of.

The leaderboard service is not restricted to use within Construct 3 - it can be used in any game engine.

Scalable and Performant

The leaderboard service scales effortlessly to millions of scores on single leaderboards with fast performance and high reliably.

Country Leaderboards Built In

Unlike other leaderboard services, you do not need to create a new leaderboard for each country you want to represent. Each leaderboard is capable of tracking countries automatically. Refer to our privacy page for details on how IP addresses and countries are calculated and stored.

Teams

Add teams into your leaderboards. Teams have their own customisable ranking system independent of the main leaderboard rankings.

Highly Customisable

Leaderboards come with a wide range of settings allowing you to customise it to your needs.

Daly, Weekly, Monthly or Annual Leaderboards

Use the `range` parameter on the get scores request to return daily, weekly, monthly or yearly leaderboards. Use the `rangeOffset` parameter to retrieve historic records.

You can also filter by country, allowing you for example to show the monthly leaderboard for the US.

Score History

Track the players score over time. You can retrieve the score history of a score up to 1 year in the past.

Compare Rank

The get score request has a `compareRanks` parameter allowing you to compare the players current rank to yesterdays rank, or any date within the last year.

Shadow Bans

If bad actors submit faked scores into your leaderboard, your able to shadow ban them based on their player ID (if one exists) and/or their IP address.

If a players score is shadow banned, it will show in the leaderboard for them - but no one else.

Auto Score Rejection

You know your game better than anyone else. You can set score thresholds to automatically reject scores that fall out of bounds. You're also able to customise how rejected scores are handled.

Store Additional Values

Each score record has the ability to record up to 3 different additional values. These are great for displaying contextual information with scores in the leaderboard - for example in a racing game you might decide to record how many gold coins were collected on the way or how many jumps you made!

Score Adjustment

You're able to adjust existing scores in the leaderboard to allow for cumulative scoring over the lifetime of the player.

Tier System

Add players into tiers based on customisable rulesets.

Many Views

The leaderboards API has many end points allowing you to:

- Get newest scores
- Get a players scores
- Get neighbour scores

And Much More

Explore the documentation to see what the leaderboard is capable of.

PRIVACY

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/privacy

Privacy of Submitted Data

When a score is submitted to the leaderboard service, the IP addresses of the score submissions are hashed with a salt before being stored in the leaderboard database. These IP addresses are **not** stored anywhere else in any other format, nor are they queried to any third party services.

Geo Location

When the score is submitted, before hashing we query a local database to ascertain the likely country the IP address originates from. The country name ascertained is stored in the score record.

The country is not in any way an accurate measurement of the users location. If for example the user is behind a VPN, the returned country will be completed inaccurate.

If you do not want to show country data in your leaderboard, there is a setting to disable it.

SHADOW BANNING SCORES

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/shadow-banned-scores

Shadow Banning Scores

There are two ways to shadow ban a score, either by their player ID or by their IP address.

When running a leaderboard service, you may find players cheat in your game or submit fake scores that pollute your leaderboard. By shadow banning the scores, the scores remain visible for the shadow banned players but do not appear for anyone else. It is an effective way to combat cheaters.

From your Game Services account pages you can shadow ban and delete scores from your leaderboard. You can also shadow ban players and IP's via our shadow ban end point.

ONE SCORE PER PLAYER

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/settings/one-score-per-player

If enabled, each player in the leaderboard will only have one score record representing their best score.

Once you set this parameter when creating a leaderboard, it will not be possible to change it at a later date.

If set, a player identifier must be sent with all scores.

This must be set if you wish to enable score history tracking and teams.

SCORE ADJUSTMENT

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/settings/score-adjustment

Leaderboard Score Adjustment

If you enabled score adjustments on your leaderboard, then the adjust score API end point will be enabled for your leaderboard.

Adjusting scores is a useful function for scores that can be cumulatively added to for the lifetime of the player across all the players sessions.

SCORE ORDERING

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/settings/score-ordering

Leaderboard Score Ordering

This simply describes how scores in a leaderboard are ordered. There are only two ways to order scores, bigger is better and smaller is better.

If you have a racing game where the fastest lap times are better, you would want to set the leaderboard score order to smaller is better.

COUNTRY RANKS

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/settings/country-rankings

Country Ranks

If you enable country rank tracking in your leaderboard, separate ranks are maintained on a country basis for scores in your leaderboard. This means you do not need a separate leaderboard for each country.

It's important to review the page on leaderboard privacy to understand how IP addresses and countries are used and stored in this service.

If you track country ranks, then you can pass a `country` parameter into various end points that allow it to return results filtered by country. For example, posting `US` into the get scores end point will return all scores where the country is US.

SCORE HISTORY

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/settings/score-history

Enabling score history for your leaderboard tracks the players score and rank for up to 365 days.

Enabling score history allows you to compare the rank and best score of a player over time.

LEADERBOARD CULTURE

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/settings/culture

Leaderboard Culture

Different cultures render numbers slightly differently - the thousands separator may not be a comma for example.

When making an API request, you can provide a culture code to use to format returned numbers. If none is provided, the leaderboards default culture code will be used.

It is recommended to pass the users culture code where possible. In Javascript, you can access this using the following code:

```
var cultureCode = navigator.language; //pt-BR
```

SCORE FORMATTING

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/settings/score-formatting

All scores are submitted as int64's. There are three types of score formats:

- Numeric
- Time
- Currency

Each score format has formatting options allowing you to customise how the formatted score is returned in the responses.

Numeric Score Format

Currency Symbol string

Appended in front of the score value, maximum length of 3 characters. With this value set to £, a score of 12 will render as "£12".

Suffix string

Appended to the end of scores, maximum length of 32 characters. With this value set to Grog Tokens, a score of 12 will render as "12 Grog Tokens".

Subunits int

The number of subunits this currency has. Most currencies have 100 subunits. Setting this to 100 and rendering a score of 500 will show as 5.00 - setting this to 0 and rendering a score of 500 will show as 500

Hide Subunits if Zero bool

If set, a score of $10.00 will render as $10

Hide thousands separator bool

If set, a score of $1,000,000 will render as $1000000/dd]

Time Score Format

The submitted score value is treated as milliseconds.

Friendly Mode bool

Renders a time of 1:30:29 as "1 hour, 30 minutes & 29 seconds".
If enabled, all other time score format options are disabled.

Score Accuracy

The accuracy to render times to. If scores never reach over 60 minutes, you can set this value to minutes.

Time Part Separator string

The symbol between parts of a time. Defaults to : - 1:02:32

Hide Milliseconds bool

Set if millisecond accuracy is not required to be rendered on scores.

Currency Score Format

The submitted score value is treated as the smallest unit of the currency. For example, if you show your scores as USD a submitted score of 123 would represent $1.23, or 123 cents.

Currency Symbol string

Appended in front of the score value, maximum length of 3 characters. With this value set to £, a score of 12 will render as "£12".

Suffix string

Appended to the end of scores, maximum length of 32 characters. With this value set to Grog Tokens, a score of 12 will render as "12 Grog Tokens".

Subunits int

The number of subunits this currency has. Most currencies have 100 subunits. Setting this to 100 and rendering a score of 500 will show as 5.00 - setting this to 0 and rendering a score of 500 will show as 500

Hide Subunits if Zero bool

If set, a score of $10.00 will render as $10

Hide Thousands Separator bool

If set, a score of $1,000,000 will render as $1000000

Custom Format

You do not need to use these score formatters. If building your own solution the score object always returns the raw score value which you can manipulate to display however you wish.

LEADERBOARD RANK TYPE

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/settings/ranking-type

Leaderboard Rank Types

There are three different ways to determine how ranks are displayed in your leaderboard.

Rank

This type will assign duplicate ranks if scores are equal. Where duplicate ranks exist, the next ranks are skipped. You can read more about this ranking method here.

Example Using Rank Type

Rank	Player	Score
1st	Tom	3,000
1st	Ash	3,000
3rd	Gordon	2,900
4rd	Piggy	2,500

Dense Rank

This type will assign duplicate ranks if scores are equal. Where duplicate ranks exist, the next ranks are **not** skipped. You can read more about this ranking method here.

Example Using Dense Rank Type

Rank	Player	Score
1st	Tom	3,000
1st	Ash	3,000
2nd	Gordon	2,900
3rd	Piggy	2,500

Row Number

This assigns a unique rank for each score. You can read more about this ranking method here.

Example Using Row Number Type

Rank	Player	Score
1st	Tom	3,000
2nd	Ash	3,000
3rd	Gordon	2,900
4th	Piggy	2,500

LEADERBOARD TEAM SETTINGS

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/settings/teams

Leaderboard Team Settings

Allow Teams

Enable or disable team functionality on this leaderboard.

Team Ranking Mode

If teams are enabled, teams will be ranked by this method. `Total scores` ranks teams with the most scores submitted highest. `Average score` ranks team with the highest average score highest. `Top score` ranks teams by the player in the team with the highest score.

LEADERBOARD SCORE REJECTION SETTINGS

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/settings/score-rejection

Leaderboard Score Rejection Settings

Reject Scores Under int64

All scores that are posted that are under this score will be rejected by the leaderboard. This value is optional.

Reject Scores Over int64

All scores that are posted that are over this score will be rejected by the leaderboard. This value is optional.

Score Rejection Mode

If set to `Shadow Ban`, players who submit scores that fall outside the permitted range will automatically be shadow banned. `Quiet fail` will cause the submitted score to appear as if it was accepted by the leaderboard, but it will not be inserted into the leaderboard. `Noisy Fail` will throw an error message if the score it outside the permitted range.

Reject Scores On Adjustment

If set, when scores are adjusted if the updated score falls outside of the acceptable range the score will then be rejected using the set rejection mode.

LEADERBOARD SECURITY SETTINGS

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/settings/security-settings

Leaderboard Security Settings

Require API Key For All POST Requests

Should all leaderboard POST requests require an API key. If this is enabled, all POST requests that don't send an API key will fail.

Require API Key For All GET Requests

Should all leaderboard GET requests require an API key. If this is enabled, all GET requests that don't send an API key will fail.

This setting is for advanced users only and you should only enable this setting if you're fully implementing the leaderboard service on your own custom back end.

TIERS

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/tiers

Leaderboard Tiers

Leaderboard tiers are an optional feature for leaderboards that allow you to group players. It is not currently possible to modify tiers through the API, it must be done through your account on Construct.net.

An example use case would be to have a `Diamond Tier` for the top 5% of scores, a `Gold Tier` for the top 15% of scores etc.

If a score belongs to a tier, it will be returned in some API objects such as the score object.

Creating a Tier

id string Required

The unique ID for this tier. This should not be displayed to players but is returned in API responses.

name string Required

A name of the tier which is displayed to players.

condition Required

The condition to specify if players belong into this tier or not.

Tier Conditions

Tiers can have one condition to specify if a score belongs in that particular tier:

Rank is exactly equal to

Matches if the players rank exactly equals the specified value.

Rank is in top N scores

If a value of 100 is provided, matches if the scores rank is <= 100

Rank is in top N% of scores

If 25 is provided, matches if the scores rank is in the top 25% of scores on the leaderboard.

Matching Scores to Tiers

When you have multiple tiers a score is iterated through all tiers and returns the tier with the first matching condition.

`Rank exactly equal to` conditions are tested first, then `Rank is in top N scores` are tested and finally `Rank is in top N% of scores` are tested.

LEADERBOARD TEAMS

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/teams

Leaderboard Teams

Leaderboard teams are an optional feature for leaderboards that allow you to assign players into teams. Teams have their own rank in your leaderboard by the ranking method specified in your leaderboard settings.

Using the get leaderboard teams API end point you can get a paginated list of all teams by rank order for your leaderboard.

You can create, rename and delete teams as well as assign players to teams, remove players from teams and delete teams using the relevant API end points.

DELETING A LEADERBOARD

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/delete-leaderboard

Deleting a Leaderboard

Deleting leaderboards must be done through your Game Services account pages.

When leaderboards are deleted, they are placed into a deletion queue and the deletion will progress in stages. It is only possible to cancel a deletion if the deletion has not yet started. Once deletion has started, it will not be possible to cancel the deletion.

THE PAGINATION OBJECT

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/objects/pagination

The Pagination Object

When a response contains or could contain multiple records, a pagination object is returned.

Example Pagination Object

```
{
    "requestedPage": 5,
    "formattedRequestedPage": "5",
    "totalPages": 1003,
    "formattedTotalPages": "1,003",
    "recordsPerPage": 2,
    "formattedRecordsPerPage": "2",
    "totalRecords": 2005,
    "formattedTotalRecords": "2,005",
    "prevPage": 4,
    "formattedPrevPage": "4"
    "nextPage": 6,
    "formattedNextPage": "6"
}
```

Object Properties

requestedPage int32

The page of results returned in this result set.

formattedRequestedPage string

The page of results returned in this result set rendered using the requested locale.

totalPages int32

The total pages of results returned in this result set.

formattedTotalPages string

The total pages of results in this result set rendered using the requested locale.

recordsPerPage int32

How many records per page are being returned in this result set.

formattedRecordsPerPage string

How many records per page are being returned in this result set rendered using the requested locale.

totalRecords int32

Total number of records in this result set.

formattedTotalRecords string

Total number of records in this result set rendered using the requested locale.

prevPage int32

The previous page number. Will not be returned in the response if this is the first page in this result set.

formattedPrevPage string

The previous page number rendered using the requested locale.
Will not be returned in the response if this is the first page in this result set.

nextPage int32

The next page number. Will not be returned in the response if this is the last page in this result set.

formattedNextPage string

The next page number rendered using the requested locale.
Will not be returned in the response if this is the last page in this result set.

THE SCORE OBJECT

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/objects/score-object

The Score Object

When you retrieve a score or list of scores, a score object will be returned.

Example Score Object

```
{
  "scoreID": "8478281d-88dd-429e-9a96-d08a3f37631c",
  "score": 4424,
  "formattedScore": "4,424",
  "rank": 15,
  "ordinal": "th",
  "formattedRank": "15th",
  "country": "NL",
  "countryRank": 3,
  "countryOrdinal": "rd",
  "formattedCountryRank": "3rd",
  "date": "2025-01-28T11:40:46.2",
  "player": { ... },
  "updates": 0,
  "compareScore": { ... },
  "teamID": "43aa1c63-28e2-464e-80f8-b1c2d8ae9696",
  "teamName": "Blue Team",
  "tier": { ... },
  "optionalValue1": 23
}
```

Object Properties

scoreID guid

A unique record ID for this score.

score int64

The stored score value.

formattedScore string

The score formatted under the score format specifications for this leaderboard, rendered using the requested locale.

rank int32

The global rank of this score

ordinal string

The ordinal for the global rank of this score.

formattedRank string

The global rank of this score rendered using the requested locale.

country string

The ISO 3166-1 alpha-2 country code of the IP address that originally posted the score. Will return as **NULL** if the country could not be ascertained.

countryRank int32

The country rank of this score. This property will not be shown if country scores are disabled in the leaderboard settings.

countryOrdinal string

The ordinal for the global country rank of this score. This property will not be shown if country scores are disabled in the leaderboard settings.

formattedCountryRank string

The global country rank of this score rendered using the requested locale. This property will not be shown if country scores are disabled in the leaderboard settings.

date datetime

The date this score was originally posted.

player playerobject

The player this score belongs to.

updates int16

How many times this score has been adjusted.

compareScore scorehistoryobject

If your get score request supports rank comparison and the compareRanks parameter is specified, a score history object for this score is returned. If there is no relevant history record for the specified time period, this property will not exist for this specific score in the response.

teamID guid

The unique ID of the team this player belongs to if they have been assigned to a team.

teamName string

The name of the team this player belongs to if they have been assigned to a team.

tier tierobject

If the player this score belongs to has been assigned to a team, the relevant team object will be returned with this score.

optionalValue `1-3` **short**

Optional values stored with the score record. They can be used to track information about a score, for example in a racing game you may wish to store how many secret coins they managed to collect.

THE LEADERBOARD OBJECT

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/objects/leaderboard

The Leaderboard Object

When you post or adjust a score, the returned leaderboard object will give more information that may be useful to show to the end user.

Example Leaderboard Object

```
{
  "globalScores": 1124005,
  "formattedGlobalScores": "1,124,005",
  "countryScores": 40104,
  "formattedCountryScores": "40,104"
}
```

Object Properties

globalScores int32

The total number of scores in the leaderboard.

formattedGlobalScores string

The total number of scores in the leaderboard rendered using the requested locale.

countryScores int32

The total number of scores in the leaderboard with the same country ID. This property will not be shown if country scores are disabled in the leaderboard settings.

formattedCountryScores string

The total number of scores in the leaderboard with the same country ID rendered using the requested locale. This property will not be shown if country scores are disabled in the leaderboard settings.

THE SCORE HISTORY OBJECT

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/objects/score-history

The Score History Object

When you retrieve the history for a score, it returns an array of score history objects. These are very similar to the score object, but contain slightly less information.

Score history objects represent a snapshot in time of the scores rankings and score value.

Example Score History Object

```
{
  "date": "2025-01-28T11:40:46.2",
  "score": 4424,
  "formattedScore": "4,424",
  "rank": 15,
  "ordinal": "th",
  "formattedRank": "15th",
  "countryRank": 3,
  "countryOrdinal": "rd",
  "formattedCountryRank": "3rd"
}
```

Object Properties

date datetime

The date of the rest of the values

score int64

The score value at this date.

formattedScore string

The score formatted under the score format specifications for this leaderboard, rendered using the requested locale.

rank int32

The global rank of this score at this date

ordinal string

The ordinal for the global rank of this score.

formattedRank string

The global rank of this score rendered using the requested locale.

countryRank int32

The country rank of this score at this date. This property will not be shown if country scores are disabled in the leaderboard settings.

countryOrdinal string

The ordinal for the global country rank of this score. This property will not be shown if country scores are disabled in the leaderboard settings.

formattedCountryRank string

The global country rank of this score rendered using the requested locale. This property will not be shown if country scores are disabled in the leaderboard settings.

THE TEAM OBJECT

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/objects/team-object

The Team Object

If teams are set up on your leaderboard some requests will return information about teams.

Example Team Object

```
{
  "teamID": "f8a013a9-1f8c-438d-bef3-4668d52b0b81",
  "dateCreated": "2025-04-02T09:09:50.657",
  "name": "Red Team",
  "rank": 2,
  "ordinal": "nd",
  "formattedRank": "2nd",
  "players": 4,
  "formattedPlayers": "4",
  "scores": 4,
  "formattedScores": "4",
  "totalScoreValues": 141116,
  "formattedTotalScoreValues": "02:21:0116",
  "averageScore": 35279,
  "formattedAverageScore": "00:35:0279",
  "bestScore": 35279,
  "formattedBestScore": "00:35:0279"
}
```

Object Properties

teamID guid

The ID of the team.

dateCreated datetime

The date the team was created.

name string

The name of the team.

rank int32

The rank of this team.

ordinal string

The ordinal for the rank of this team.

formattedRank string

The rank of this score rendered using the requested locale.

players int32

The number of players assigned to this team.

formattedPlayers string

The number of players assigned to this team rendered using the requested locale.

scores int32

The number of scores posted in this team.

formattedScores string

The number of scores posted in this team rendered using the requested locale.

totalScoreValues decimal

The sum value of all scores posted in this team.

formattedTotalScoreValues string

The sum value of all scores posted in this team formatted under the score format specifications for this leaderboard, rendered using the requested locale.

averageScore int64

The average of all the scores posted in this team.

formattedAverageScore string

The average of all the scores posted in this team formatted under the score format specifications for this leaderboard, rendered using the requested locale.

bestScore int64

The best score posted in this team.

formattedBestScore string

The best score posted in this team formatted under the score format specifications for this leaderboard, rendered using the requested locale.

THE TIER OBJECT

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/objects/tier-object

The Tier Object

If tiers are set up on the leaderboard, scores will return a matching tier indicating what current tier this score belongs to.

Example Tier Object

```
{
  "id": "diamond",
  "name": "Diamond Tier"
}
```

Object Properties

id string

A unique ID for this tier for your leaderboard.

name string

The name of the tier that can be shown to players.

THE SHADOW BAN OBJECT

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/objects/shadow-ban-object

The Shadow Ban Object

Returned when requesting lists of shadow bans for a specified leaderboard. Depending on if you query player identifier shadow bans or IP address shadow bans will determine the properties of this object that are returned.

Player ID Shadow Ban Object

When querying player ID shadow bans, the following shadow ban object is returned.

```
{
  "dateBanned": "2025-04-23T00:00:00",
  "player": { ... }
}
```

Player ID Shadow Ban Properties

dateBanned datetime

The date the ban was created.

player playerobject

The player that is shadow banned.

IP Shadow Ban Object

When querying IP shadow bans, the following shadow ban object is returned.

```
{
  "dateBanned": "2025-04-23T00:00:00",
  "ipHash": -23471271,
  "country": "US"
}
```

IP Shadow Ban Properties

dateBanned datetime

The date the ban was created.

ipHash int

The hashed IP address of the IP that is shadow banned.

country string

If the country of the underlying IP is known, the ISO 3166-1 alpha-2 country code for the IP will be shown in this property.

POST A NEW SCORE

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/post-new-score

Post a new score

Records a new score record into a leaderboard.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/postscore.json
```

Authenticating The Request

There are two ways to authenticate this request. One with a **secret**, the other with a **sessionKey** and **hash**.

Secret Key Authentication

secret string

Your games secret key.

Session Key Authentication

sessionKey string

The session key of the player you're making the request against.

hash string

If the game the leaderboard is associated with does not have a secret key, then a hash for the post score request must be provided.
A hash is generated as the SHA256 of a string of values in the request:

```
var key = (leaderboardID + "." + score + "." + unixTimestamp).Normalize();
return SHA256(key);
```

Player identifier is an optional value, this should be an empty string if not used but the dot after the timestamp is still required.

Authenticating post score requests with a hash makes it harder for casual request interception and manipulation by players.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you want to post this score to

score int64 Required

The score you're posting. The maximum value of a score is **9223372036854775807** and the minimum value is **-9223372036854775808**.

timestamp int64 Required

The unix timestamp of the score. Adding historic scores or scores for future dates is not supported. This timestamp should be the date the request to the post score API was generated. This value is required as it is used in the request verification.

requesterIP string

If a secret key is being used, this value is required and represents the IP address of the client the score belongs to.

playerID Guid

The ID of the player this score is for. Refer to the authentication service for how to retrieve this ID.

culture string

Optionally specify the locale to render returned values with. If no value or an invalid value is provided, the leaderboard's default culture code is used.

opt1 short

An optional value to store with this score record.

opt2 short

An optional value to store with this score record.

opt3 short

An optional value to store with this score record.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"timestamp is required parameter.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-

attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true,
  "formattingCulture": "en-US",
  "score": { ... },
  "leaderboard": { ... },
  "personalBest": true
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

formattingCulture string

The locale used to render various formatted values in the response. This will fall back to the leaderboard's default locale if no culture value is posted or the posted culture value is invalid.

score score

A score object containing data about the new score.

leaderboard leaderboard

A leaderboard object containing useful contextual information.

personalBest bool

Is the adjusted or newly posted score this players personal best? This property does not appear if no player ID is saved with the score.

ADJUST EXISTING SCORE

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/adjust-score

Adjust existing score

Adjusts the score value of an existing score in the leaderboard. You can pass a positive value to increase the value of the score, or a negative value to decrease it.

A leaderboard must permit score adjustments in it's settings otherwise calls to adjust scores will fail.

Adjusting a score will not update the date of the score, but will increment it's **updates** counter.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/adjustscore.json
```

Authenticating The Request

There are two ways to authenticate this request. One with a **secret**, the other with a **sessionKey** and **hash**.

Secret Key Authentication

secret string

Your games secret key.

Session Key Authentication

sessionKey string

The session key of the player you're making the request against.

hash string

If the game the leaderboard is associated with does not have a secret key, then a hash for the increment score request must be provided.
A hash is generated as the SHA256 of a string of values in the request:

```
var key = (leaderboardID + "." + adjustment + "." + unixTimestamp).Normalize();
return SHA256(key);
```

Authenticating increment score requests with a hash makes it harder for casual request interception and manipulation by players.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you want to increment the score of

adjustment int64 Required

The value to modify the existing score by. The maximum value of a score is **9223372036854775807** and the minimum value is **-9223372036854775808**. If an adjustment results in a value that will exceed these ranges an error will be returned.

timestamp int64 Required

The unix timestamp of the date of the request. This value is required as it is used in the request verification.

scoreID guid

The ID of the score record you wish to adjust the score for.
Either scoreID or playerID must be specified.

playerID Guid

The ID of the player to adjust the score for. Either playerID or scoreID must be specified. If there are multiple scores for the passed playerID then this players best score will be adjusted. Refer to the authentication service for how to retrieve this ID.

culture string

Optionally specify the locale to render returned values with. If no value or an invalid value is provided, the leaderboard's default culture code is used.

opt1 short

Adjust the optional value 1 value by this amount. If the existing value is not set, will set the value of the score record to this value.

opt2 short

Adjust the optional value 2 value by this amount. If the existing value is not set, will set the value of the score record to this value.

opt3 short

Adjust the optional value 3 value by this amount. If the existing value is not set, will set the value of the score record to this value.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Either hash or secret must be posted.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true,
  "formattingCulture": "en-US",
  "score": { ... },
  "leaderboard": { ... },
  "personalBest": true
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

formattingCulture string

The locale used to render various formatted values in the response. This will fall back to the leaderboard's default locale if no culture value is posted or the posted culture value is invalid.

score score

A score object containing data about the updated score.

leaderboard leaderboard

A leaderboard object containing useful contextual information.

personalBest bool

Is the adjusted or newly posted score this players personal best? This property does not appear if no player ID is saved with the score.

GET SCORES

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/get-scores

Get newest posted scores

Return paginated results of scores. Various filters are supported allowing you to:

- Return the all time scores for a leaderboard
- Return country specific all time scores for a leaderboard
- Return daily, weekly, monthly or yearly scores for a leaderboard for all countries or specific countries

Request URL

All parameters in the request can be sent in the querystring or posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/getscores.json
```

Authenticating The Request

No authentication is required for this request type.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you are querying

perPage int32

How many scores to display on each page of results, from 1 to 500. If an invalid value is provided the default value of 20 is used.

page int32

What page of results to return. If no value is provided, will default to the first page. If the value exceeds the total pages, the last page will be returned.

---

country string

The ISO 3166-1 alpha-2 country code you wish to filter results by. If you wish to retrieve results with no known country you can pass the value `xx`.

---

range string

Either `Daily`, `Weekly`, `Monthly` or `Yearly` can be provided. If specified, will filter the results by this time period. Weekly leaderboards run Monday to Sunday.

---

rangeOffset int32

If **range** is specified, you can offset the returned results by this amount. For example, a range of **daily** with an offset of 1 will return yesterdays daily leaderboard. A range of **monthly** with a range of 5 will return the monthly leaderboard from 5 months ago.

---

compareRanks int

Optionally specify this value for leaderboards that support score history. This value represents how many days ago you wish to compare returned scores in this query to. For example, if 10 is specified then the relevant score history object will be returned in the score object representing what this score was 10 days ago.

---

requesterPlayerID Guid

If the player ID is known, pass their player ID here. If any of this players scores are shadow banned they will show in the response. Refer to the authentication service for how to retrieve this ID.

---

requesterIP string

If not passed, the IP address of the request origin will be used. You should pass the visitors IP address with this parameter if you're not querying the end point through a client side implementation, otherwise this IP's shadow banned scores will not show.

---

culture string

Optionally specify the locale to render returned values with. If no value or an invalid value is provided, the leaderboard's default culture code is used.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"leaderBoardID is required parameter.",
  "shouldRetry":false
}
```

---

success bool

If the request was successful or not. For request failures, this will always be false.

---

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

---

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true,
  "pagination": { ... },
  "formattingCulture": "en-US",
  "scores": [...],
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

pagination pagination

A pagination object that helps you browse through pages of results.

formattingCulture string

The locale used to render various formatted values in the response. This will fall back to the leaderboard's default locale if no culture value is posted or the posted culture value is invalid.

scores array

An array of score objects for this page of results.

GET NEWEST SCORES

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/get-newest-scores

Get newest posted scores

Returns scores in the leaderboard, newest first.

Request URL

All parameters in the request can be sent in the querystring or posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/getnewestscores.json
```

Authenticating The Request

No authentication is required for this request type.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you are querying

perPage int32

How many scores to display on each page of results, from 1 to 500. If an invalid value is provided the default value of 20 is used.

page int32

What page of results to return. If no value is provided, will default to the first page. If the value exceeds the total pages, the last page will be returned.

country string

The ISO 3166-1 alpha-2 country code you wish to filter results by. If you wish to retrieve results with no known country you can pass the value `xx`.

---

requesterPlayerID Guid

If the player ID is known, pass their player ID here. If any of this players scores are shadow banned they will show in the response. Refer to the authentication service for how to retrieve this ID.

---

requesterIP string

If not passed, the IP address of the request origin will be used. You should pass the visitors IP address with this parameter if you're not querying the end point through a client side implementation, otherwise this IP's shadow banned scores will not show.

---

culture string

Optionally specify the locale to render returned values with. If no value or an invalid value is provided, the leaderboard's default culture code is used.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"leaderBoardID is required parameter.",
  "shouldRetry":false
}
```

---

success bool

If the request was successful or not. For request failures, this will always be false.

---

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true,
  "pagination": { ... },
  "formattingCulture": "en-US",
  "scores": [...],
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

pagination pagination

A pagination object that helps you browse through pages of results.

formattingCulture string

The locale used to render various formatted values in the response. This will fall back to the leaderboard's default locale if no culture value is posted or the posted culture value is invalid.

scores array

An array of score objects for this page of results.

GET PLAYER SCORES

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/get-player-scores

Get player scores

Given a player ID, return all this players scores in the leaderboard, best first. Where a leaderboard only supports 1 score per player ID, only one result can ever be returned.

Request URL

All parameters in the request can be sent in the querystring or posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/getplayerscores.json
```

Authenticating The Request

No authentication is required for this request type.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you are querying

playerID Guid Required

The player ID to return scores for. Refer to the authentication service for how to retrieve this ID.

perPage int32

How many scores to display on each page of results, from 1 to 500. If an invalid value is provided the default value of 20 is used.

page int32

What page of results to return. If no value is provided, will default to the first page. If the value exceeds the total pages, the last page will be returned.

culture string

Optionally specify the locale to render returned values with. If no value or an invalid value is provided, the leaderboard's default culture code is used.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"No playerID value was sent",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true,
  "pagination": { ... },
  "formattingCulture": "en-US",
  "scores": [...],
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

pagination pagination

A pagination object that helps you browse through pages of results.

formattingCulture string

The locale used to render various formatted values in the response. This will fall back to the leaderboard's default locale if no culture value is posted or the posted culture value is invalid.

scores array

An array of score objects for this page of results.

GET SCORE NEIGHBOURS

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/get-score-neighbours

Get neighbour scores

Given a player ID or a score record ID, get the surrounding scores in the leaderboard.

Request URL

All parameters in the request can be sent in the querystring or posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/getscoreneighbours.json
```

Authenticating The Request

No authentication is required for this request type.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you are querying

playerID Guid

The player ID to query. Either playerID or scoreID must be specified in the request. Refer to the authentication service for how to retrieve this ID.

scoreID guid

The unique record ID of the score to query. Either scoreID or playerID must be specified in the request.

range int32

How many scores to return on either side of the queried score.
If no value is specified, or an invalid value is specified this defaults to 5 which will return up to 11 scores (the queried score, plus up to 5 scores on each side).

---

compareRanks int

Optionally specify this value for leaderboards that support score history. This value represents how many days ago you wish to compare returned scores in this query to. For example, if 10 is specified then the relevant score history object will be returned in the score object representing what this score was 10 days ago.

---

requesterIP string

If not passed, the IP address of the request origin will be used. You should pass the visitors IP address with this parameter if you're not querying the end point through a client side implementation, otherwise this IP's shadow banned scores will not show.

---

culture string

Optionally specify the locale to render returned values with. If no value or an invalid value is provided, the leaderboard's default culture code is used.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Invalid leaderboardID",
  "shouldRetry":false
}
```

---

success bool

If the request was successful or not. For request failures, this will always be false.

---

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

---

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true,
  "formattingCulture": "en-US",
  "scores": [...],
}
```

---

success bool

If the request was successful or not. For request successes, this will always be true.

---

formattingCulture string

The locale used to render various formatted values in the response. This will fall back to the leaderboard's default locale if no culture value is posted or the posted culture value is invalid.

---

scores array

An array of score objects. The score array will contain the requested score, with up to `range` neighbours before and after it.

GET SCORE HISTORY

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/score-history

Get score history

Given a player ID or a score record ID, return the daily history of this score for the last 365 days.

This is only supported on leaderboards where one score per player identifier is set.

If the rankings and score value do not change day to day then no history record will exist - there will be a gap in the records.

Request URL

All parameters in the request can be sent in the querystring or posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/getscorehistory.json
```

Authenticating The Request

No authentication is required for this request type.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you are querying

playerID guid

The player ID to query. Either playerID or scoreID must be specified in the request. Refer to the authentication service for how to retrieve this ID.

---

scoreID guid

The unique record ID of the score to query. Either scoreID or playerID must be specified in the request.

---

culture string

Optionally specify the locale to render returned values with. If no value or an invalid value is provided, the leaderboard's default culture code is used.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"This leaderboard does not support score histories.",
  "shouldRetry":false
}
```

---

success bool

If the request was successful or not. For request failures, this will always be false.

---

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

---

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true,
  "scoreID": "c9547c1e-d219-4894-8c9f-13c5119c4563",
  "player": { ... },
  "country": "US",
  "formattingCulture": "en-US",
  "scoreHistory ": [...],
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

scoreID guid

The unique record ID for this score

player playerobject

The player of this score record

country string

The ISO 3166-1 alpha-2 country code of the IP address that originally posted the score. Will return as **NULL** if the country could not be ascertained.

formattingCulture string

The locale used to render various formatted values in the response. This will fall back to the leaderboard's default locale if no culture value is posted or the posted culture value is invalid.

scoreHistory array

An array of up to 365 score history objects, ordered by oldest score first.

DELETE SCORES

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/delete-scores

Delete Scores

Deletes scores either by specific score ID, or scores by a player ID.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/deletescores.json
```

Authenticating The Request

A **secret** must be passed in this request. This end point should not be called client side ever as it would expose the secret key.

secret string

If the game the leaderboard is associated with has a secret key, this must be provided or the request will be rejected.
Secret keys must never be exposed to clients.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you want to post this score to

playerID guid

The ID of the player you want to delete scores for. The scores will be deleted in a random order. A playerID or scoreID must be provided in the request. Refer to the authentication service for how to retrieve this ID.

scoreID guid

The ID of the score to delete. A scoreID or playerID must be provided in the request.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"timestamp is required parameter.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true,
  "scoresDeleted": 500,
  "mightHaveMore": true
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

scoresDeleted int32

The number of scores deleted. This can range from 0 to 500. No more than 500 scores can be deleted in any one request.

mightHaveMore bool

If the scores deleted is the maximum allowed to be deleted in one request, this value will show as true indicating that there may be more scores to delete in this request.

SHADOW BAN PLAYERS

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/shadow-ban

Shadow Ban Players

You can shadow ban IP addresses and player identifiers from your leaderboards. This will still show their posted scores to the affected player identifiers and IP addresses, but will be hidden from view for everyone else viewing your leaderboard.

IP addresses are never stored in their raw format in our database, see our privacy page for more information.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/shadowban.json
```

Authenticating The Request

A **secret** must be passed in this request. This end point should not be called client side ever as it would expose the secret key.

secret string

If the game the leaderboard is associated with has a secret key, this must be provided or the request will be rejected.
Secret keys must never be exposed to clients.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you want to add to the shadow ban list

playerID guid

The ID of the player you want to shadow ban. A playerID, ipAddress, scoreID or ipHash must be provided in the request. Refer to the authentication service for how to retrieve this ID.

ipAddress string

The IP address to shadow ban. Can be an IPV4 or IPV6 address, ranges are not permitted. A playerID, ipAddress, scoreID or ipHash must be provided in the request.

scoreID guid

The ID of the score to shadow ban. This bans the scores player identifier AND IP address. A playerID, ipAddress, scoreID or ipHash must be provided in the request.

ipHash int

The IP hash to shadow ban. A playerID, ipAddress, scoreID or ipHash must be provided in the request.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"timestamp is required parameter.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

---

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true
}
```

---

success bool

If the request was successful or not. For request successes, this will always be true.

REMOVE SHADOW BAN

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/remove-shadow-ban

Remove Shadow Ban

You can remove IP addresses and player identifiers from your leaderboards shadow ban lists.

IP addresses are never stored in their raw format in our database, see our privacy page for more information.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/unshadowban.json
```

Authenticating The Request

A **secret** must be passed in this request. This end point should not be called client side ever as it would expose the secret key.

secret string

If the game the leaderboard is associated with has a secret key, this must be provided or the request will be rejected.
Secret keys must never be exposed to clients.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you want to add to the shadow ban list

playerID guid

The ID of the player you want to remove from this leaderboards shadow ban list. A playerID, ipAddress, scoreID or ipHash must be provided in the request. Refer to the authentication service for how to retrieve this ID.

---

ipAddress string

The IP address to remove from this leaderboards shadow ban list. Can be an IPV4 or IPV6 address, ranges are not permitted. A playerID, ipAddress, scoreID or ipHash must be provided in the request.

---

scoreID guid

The ID of the score to remove from the shadow ban list. This unbans the scores player identifier AND IP address. A playerID, ipAddress, scoreID or ipHash must be provided in the request.

---

ipHash int

The IP hash to remove from the shadow ban list. A playerID, ipAddress, scoreID or ipHash must be provided in the request.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"timestamp is required parameter.",
  "shouldRetry":false
}
```

---

success bool

If the request was successful or not. For request failures, this will always be false.

---

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

GET IP SHADOW BANS

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/get-ip-shadow-bans

Get IP Shadow Bans

Retrieve a list of all IP hashes that have been shadow banned.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/getipshadowbans.json
```

Authenticating The Request

A **secret** must be passed in this request. This end point should not be called client side ever as it would expose the secret key.

secret string

If the game the leaderboard is associated with has a secret key, this must be provided or the request will be rejected.
Secret keys must never be exposed to clients.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you are querying

perPage int32

How many ban records to display on each page of results, from 1 to 500. If an invalid value is provided the default value of 20 is used.

page int32

What page of results to return. If no value is provided, will default to the first page. If the value exceeds the total pages, the last page will be returned.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"This leaderboard does not support score histories.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true,
  "pagination": { ... },
  "bans": [...],
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

pagination pagination

A pagination object that helps you browse through pages of results.

bans shadowBan

An array of shadow ban objects for this page of results.

GET PLAYER IDENTIFIER SHADOW BANS

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/get-player-shadow-bans

Get Player Identifier Shadow Bans

Retrieve a list of all player identifiers that have been shadow banned.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/getplayeridshadowbans.json
```

Authenticating The Request

A **secret** must be passed in this request. This end point should not be called client side ever as it would expose the secret key.

secret string

If the game the leaderboard is associated with has a secret key, this must be provided or the request will be rejected.
Secret keys must never be exposed to clients.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you are querying

perPage int32

How many ban records to display on each page of results, from 1 to 500. If an invalid value is provided the default value of 20 is used.

page int32

What page of results to return. If no value is provided, will default to the first page. If the value exceeds the total pages, the last page will be returned.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"This leaderboard does not support score histories.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true,
  "pagination": { ... },
    "bans": [...],
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

pagination pagination

A pagination object that helps you browse through pages of results.

bans shadowBan

An array of shadow ban objects for this page of results.

CREATE A LEADERBOARD TEAM

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/teams/create-team

Create a Leaderboard Team

Create a new team in a leaderboard.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/createteam.json
```

Authenticating The Request

A **secret** must be passed in this request. This end point should not be called client side ever as it would expose the secret key.

secret string

If the game the leaderboard is associated with has a secret key, this must be provided or the request will be rejected.
Secret keys must never be exposed to clients.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you want to create a team for.

teamName string Required

The name of the team. Must not exceed 64 characters in length.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Teams are not enabled for this leaderboard.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true,
  "id": "52bb3e1e-2620-4782-81ce-f22c2d973478"
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

id guid

The ID of the new team.

RENAME LEADERBOARD TEAM

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/teams/rename-team

Rename a Leaderboard Team

Rename an existing leaderboard team.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/renameteam.json
```

Authenticating The Request

A **secret** must be passed in this request. This end point should not be called client side ever as it would expose the secret key.

secret string

If the game the leaderboard is associated with has a secret key, this must be provided or the request will be rejected.
Secret keys must never be exposed to clients.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you want to create a team for.

teamID guid Required

The ID of the team you're renaming.

teamName string Required

The new name of the team. Must not exceed 64 characters in length.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Team name is too long.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

ASSIGN A PLAYER TO A TEAM

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/teams/assign-player

Assign a Player to a Team

Assign a player identifier to a team.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/assignplayertoteam.json
```

Authenticating The Request

A **secret** must be passed in this request. This end point should not be called client side ever as it would expose the secret key.

secret string

If the game the leaderboard is associated with has a secret key, this must be provided or the request will be rejected.
Secret keys must never be exposed to clients.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you want to create a team for.

teamID guid Required

The ID of the team you're renaming.

playerID guid Required

The player ID you're adding to the team. Refer to the authentication service for how to retrieve this ID.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"That player is already on this team.",
  "shouldRetry":false
}
```

---

success bool

If the request was successful or not. For request failures, this will always be false.

---

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

---

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true
}
```

---

success bool

If the request was successful or not. For request successes, this will always be true.

REMOVE PLAYER FROM TEAM

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/teams/remove-player

Remove a Player from a Team

Remove a player from a leaderboard team.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/removeplayerfromteam.json
```

Authenticating The Request

A **secret** must be passed in this request. This end point should not be called client side ever as it would expose the secret key.

secret string

If the game the leaderboard is associated with has a secret key, this must be provided or the request will be rejected.
Secret keys must never be exposed to clients.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you want to create a team for.

teamID guid Required

The ID of the team you're renaming.

playerID guid Required

The player identifier you're removing from the team. Refer to the authentication service for how to retrieve this ID.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"That player isn't on this team.",
  "shouldRetry":false
}
```

---

success bool

If the request was successful or not. For request failures, this will always be false.

---

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

---

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true
}
```

---

success bool

If the request was successful or not. For request successes, this will always be true.

DELETE A TEAM

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/teams/delete-team

Delete a Team

Delete a team from the leaderboard.

Request URL

All parameters in the request must be posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/deleteteam.json
```

Authenticating The Request

A **secret** must be passed in this request. This end point should not be called client side ever as it would expose the secret key.

secret string

If the game the leaderboard is associated with has a secret key, this must be provided or the request will be rejected.
Secret keys must never be exposed to clients.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you want to create a team for.

teamID guid Required

The ID of the team you're deleting.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Passed team ID does not exist.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

GET TEAM

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/teams/get-team

Get Teams

Retrieve a single team in a leaderboard.

Request URL

All parameters in the request can be sent in the querystring or posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/getteam.json
```

Authenticating The Request

No authentication is required for this request type.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you want to fetch the teams for.

teamID guid Required

The ID of the team you wish to fetch.

culture string

Optionally specify the locale to render returned values with. If no value or an invalid value is provided, the leaderboard's default culture code is used.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Invalid leaderboard ID passed.",
  "shouldRetry":false
}
```

---

success bool

If the request was successful or not. For request failures, this will always be false.

---

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

---

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true
  "team": { ... },
  "formattingCulture": "en-US"
}
```

---

success bool

If the request was successful or not. For request successes, this will always be true.

---

team teamobiect

team teamObject

The team object of the queried team.

formattingCulture string

The locale used to render various formatted values in the response. This will fall back to the leaderboard's default locale if no culture value is posted or the posted culture value is invalid.

GET TEAMS

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/teams/get-teams

Get Teams

Retrieve all teams in a leaderboard in their rank order.

Request URL

All parameters in the request can be sent in the querystring or posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/getteams.json
```

Authenticating The Request

No authentication is required for this request type.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you want to fetch the teams for.

perPage int32

How many teams to display on each page of results, from 1 to 100. If an invalid value is provided the default value of 20 is used.

page int32

What page of results to return. If no value is provided, will default to the first page. If the value exceeds the total pages, the last page will be returned.

culture string

Optionally specify the locale to render returned values with. If no value or an invalid value is provided, the leaderboard's default culture code is used.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"Invalid leaderboard ID passed.",
  "shouldRetry":false
}
```

---

success bool

If the request was successful or not. For request failures, this will always be false.

---

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

---

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true
  "pagination": { ... },
  "teams": [ ... ],
  "formattingCulture": "en-US"
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

pagination pagination

A pagination object that helps you browse through pages of results.

teams teamobject

A list of team objects.

formattingCulture string

The locale used to render various formatted values in the response. This will fall back to the leaderboard's default locale if no culture value is posted or the posted culture value is invalid.

GET TEAM PLAYERS

View online: https://www.construct.net/en/game-services/manuals/game-services/leaderboards/api-end-points/teams/team-players

Get Team Players

Retrieve all players in a team.

Request URL

All parameters in the request can be sent in the querystring or posted. Make all requests to the following URL:

```
https://leaderboards.construct.net/getteamplayers.json
```

Authenticating The Request

No authentication is required for this request type.

Request Parameters

leaderboardID guid Required

The ID of the leaderboard you want to fetch the teams for.

teamID guid Required

The ID of the team you want to fetch the players for.

perPage int32

How many teams to display on each page of results, from 1 to 100. If an invalid value is provided the default value of 20 is used.

page int32

What page of results to return. If no value is provided, will default to the first page. If the value exceeds the total pages, the last page will be returned.

order string

If this value equals `score` players will be returned, best scores first. Any other value will return the players alphabetically.

culture string

Optionally specify the locale to render returned values with. If no value or an invalid value is provided, the leaderboard's default culture code is used.

Failure Response

Unsuccessful responses always return 4xx HTTP status codes.

```
{
  "success":false,
  "errorMessage":"That team does not exist.",
  "shouldRetry":false
}
```

success bool

If the request was successful or not. For request failures, this will always be false.

errorMessage string

An short message explaining why the request was denied.
This should probably not be shown to clients.

shouldRetry bool

If true, this means the request is valid but it couldn't be processed at this current time - usually due to rate limits.
If this value returns as true, it's recommended to wait a few seconds then re-attempt the request. When re-attempting requests, make sure you regenerate the **timestamp** and **hash** parameters if no secret is being used.

Success Response

Successful responses always return the HTTP 200 status code.

```
{
  "success": true
  "pagination": { ... },
  "team": { ... },
  "players": [
    {
      "player": { ... },
      "currentScore": 35279,
      "formattedScore": "00:35:0279"
    },
    {
      "player": { ... },
      "currentScore": 35279,
      "formattedScore": "00:35:0279"
    },
    {
      "player": { ... },
      "currentScore": null
    },
    {
      "player": { ... },
      "currentScore": 35279,
      "formattedScore": "00:35:0279"
    }
  ],
  "formattingCulture": "en-US"
}
```

success bool

If the request was successful or not. For request successes, this will always be true.

pagination pagination

A pagination object that helps you browse through pages of results.

team teamobject

A team object for this team.

players object

A list of players with their best score.

players.player playerobject

The player in this team.

players.currentScore int64

This players current best score. Will show as `null` if they do not currently have any posted scores.

players.formattedScore int64

This players current best score formatted under the score format specifications for this leaderboard, rendered using the requested locale.

formattingCulture string

The locale used to render various formatted values in the response. This will fall back to the leaderboard's default locale if no culture value is posted or the posted culture value is invalid.