# Overview

Copy Page

- Introduction
  - Workflow
    - Game flow
    - Freespins bonus flow
    - Free spins bonus cancelation
    - Bonus Buy flow
    - The dependency of free spin issuance on bet levels
    - Promo flow
      - Promo debit
      - Promo win
      - Promo prize
      - Promo rollback
  - How to launch games
    - Prerequisites
    - Steps
  - FAQ
  - API Protocol
  - Security
    - Generating X-Request-Sign in Go
    - Generating X-Request-Sign in Ruby
    - Generating X-Request-Sign in Java
    - Generating X-Request-Sign in JS
    - Generating X-Request-Sign in Postman
  - Data format
    - Money Amount Format
    - Currency format
    - Country format
    - Jurisdiction format
    - Locale format
    - Time Format
    - URL format
  - Errors Format
    - Error Examples
    - Available error API codes
  - Gamelist Format

## Introduction

An *internal* casino (referred to as "Casino" further) is our product (a.k.a., a tenant) that works directly with players.

An *external* provider (referred to as "Provider" further) is a final game content provider.

***01.tech Aggregator*** is an *internal* game aggregator (referred to as "Game Aggregator" further). It acts as an agent between our *internal* casinos and *external* game providers.

### Workflow

Aggregator API is divided into several parts:

- Communication between Game Aggregator and Provider:
  - Aggregator API for Provider - describes requests from Provider to Game Aggregator.
  - Provider API for Aggregator - describes requests from Game Aggregator to Provider.
- Communication between Casino and Game Aggregator:
  - Aggregator API for Casino - describes requests from Casino to Game Aggregator.
  - Casino API for Aggregator - describes requests from Game Aggregator to Casino.

#### Game flow

Open game flow:

1. Player opens a game page in the Casino.
2. The Casino sends a Launcher/Real OR Launcher/Demo request to Game Aggregator (Game Aggregator passes that request to the Provider).
3. The Provider responds with a **Launch URL** to Game Aggregator (Game Aggregator passes that response to the Casino).
4. The Casino uses the **Launch URL** to open the game in an `iframe`.
5. The player starts playing.

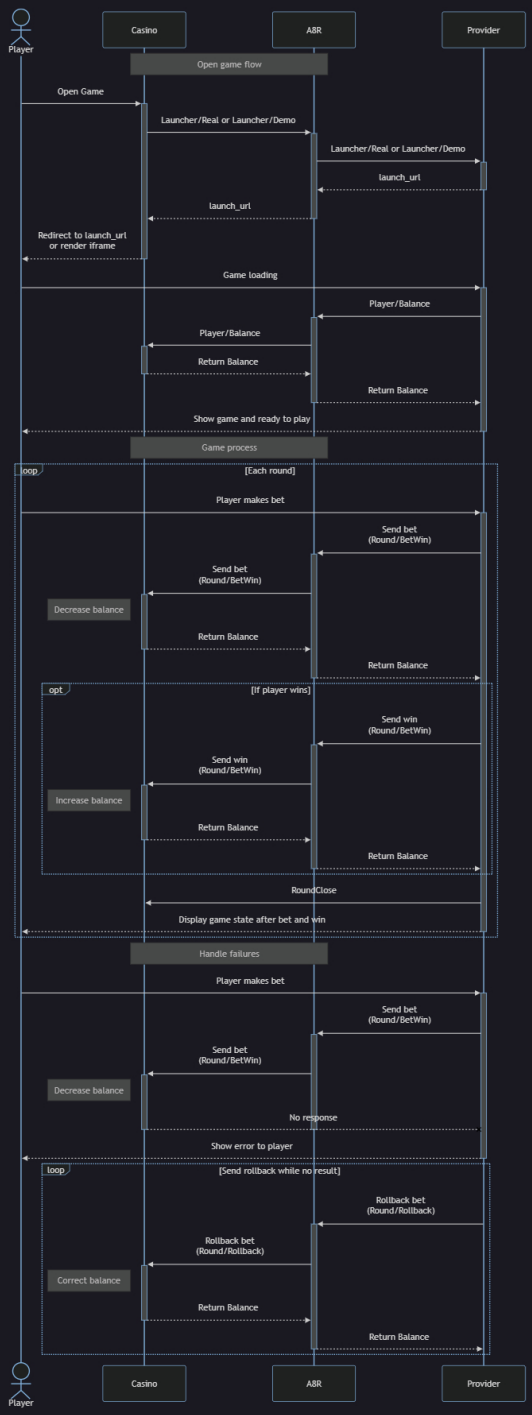
Game Process:

1. An event occurs in the game (such as getting the balance, betting, winning, or finishing a round).
2. The Provider should send a corresponding request (Player/Balance, Round/BetWin, Round/Rollback, Round/Finish) to Game Aggregator.
3. Game Aggregator sends the request to the Casino.
4. The Casino handles the event and responds to Game Aggregator.

Introduction
Workflow
Game flow
Freespins bonus flow
Free spins bonus cancelation
The dependency of free spin issuanc...
Bonus Buy flow
Bonus Types
Casino-Side Initiated Bonus Buy Flow:
Aggregator-Side Initiated Bonus Buy...
Game Process:
Bonus Buy Cancelation
How to launch games
Prerequisites
Steps
Scenario 1: Launching via URL (Sta...
Scenario 2: Launching via Our Con...
Promo flow
Promo debit
Promo win
Promo prize
Promo rollback
FAQ
API Protocol
Security
Generating signature in Go
Generating signature in Ruby
Generating signature in Java
Generating signature in JS
Generating signature in Postman
Data format
Money Amount Format

5. Game Aggregator resends the Casino response to the Provider.
6. The player continues to play.

Find the more detailed workflow diagram below:




### Freespins bonus flow

The free spins bonus is a promotional feature allowing a Casino to reward a Player with a limited number of free spins. The process for issuing free spins can be initiated from the Casino side or from the Aggregator side.

Casino-Side Initiated Free Spins Flow:

1. The Casino rewards the Player with bonus free spins for a list of games, which can include one or many. The bonus can only be played on any game from the list.
2. The Player activates the free spin bonus.
3. The Casino sends a request for Freespins/Issue to Game Aggregator (which then passes the request to the Provider).
4. The Provider confirms the issuance of the free spins.
5. The Casino notifies the Player that the bonus is activated.
6. The Player starts playing any game from the list.

Aggregator-Side Initiated Free Spins Flow:

1. The Casino rewards the Player with bonus free spins by creating a free spins campaign via the Aggregator's user interface. The campaign includes a list of one or more eligible games.
2. The Aggregator sends a Freespins/Issue request to the Game Provider.

3. The Game Provider confirms the successful issuance of the free spins.
4. The Aggregator sends a notification Freespins/Issue about the issued free spins to the Casino.
5. The Casino notifies the Player that the bonus has been activated.
6. The Player starts playing any game from the approved list.

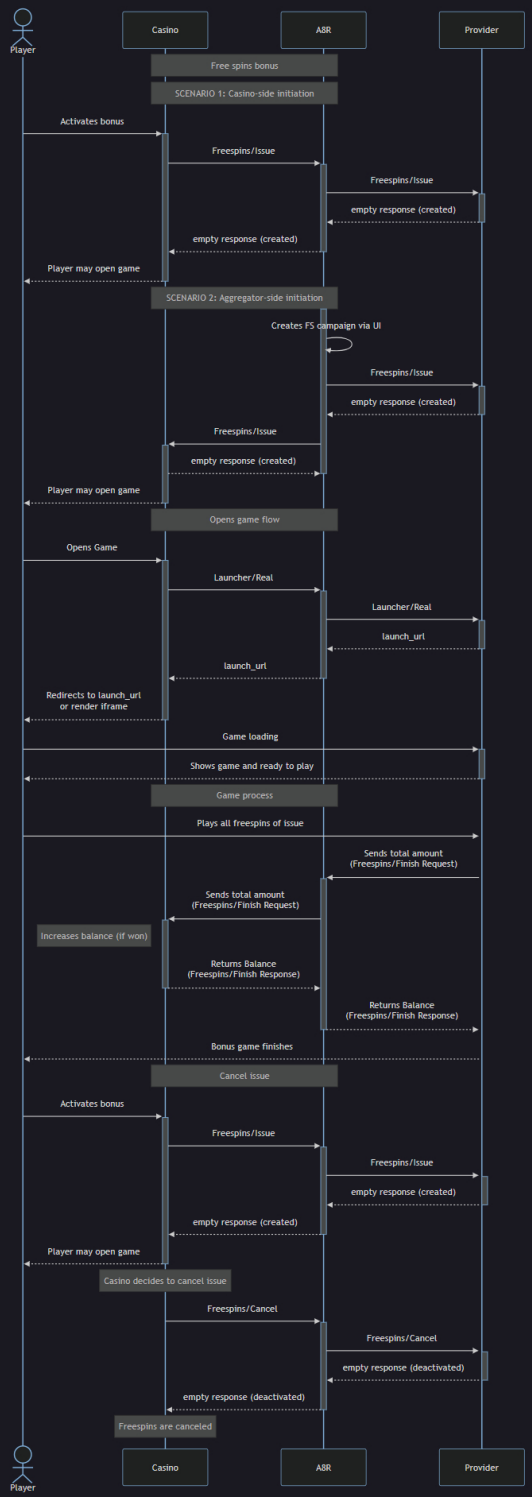
Game Process:

1. The free spins bonus game is not related to the Player's actual balance in the Casino. Instead, the Player sees the number of free spins that have been issued.
2. As the Player makes a spin, the number of free spins decreases.
3. The total amount of winnings is calculated and summed up on the Provider's side.
4. When all the free spins have been played, the Provider sends a request for Freespins/Finish to Game Aggregator (which then passes Freespins/Finish request to the Casino) with the total amount of winnings.

### Free spins bonus cancelation

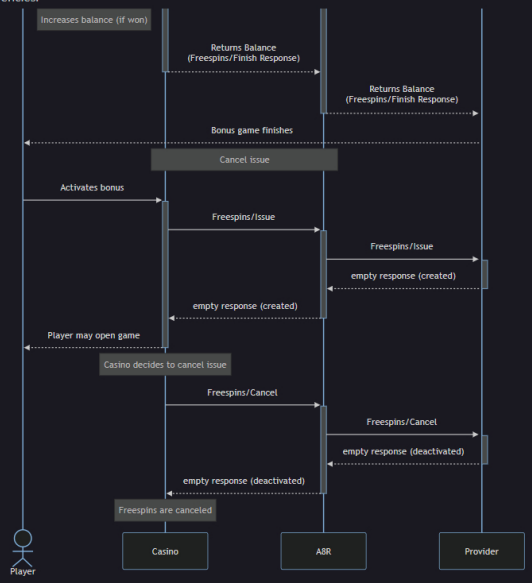
The Casino can cancel the free spins bonus if it hasn't expired yet, if the Player hasn't started playing the bonus game, or is currently playing with the bonus.
This can be done by sending a Freespins/Cancel request.

Find the more detailed workflow diagram below:



## The dependency of free spin issuance on bet levels

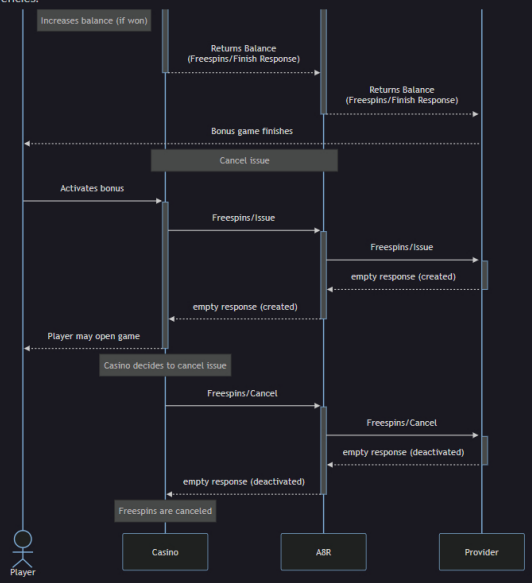
1. `bet_levels` is a set of rules (restrictions) for configuring free spins with regard to different currencies.





## The dependency of free spin issuance on bet levels

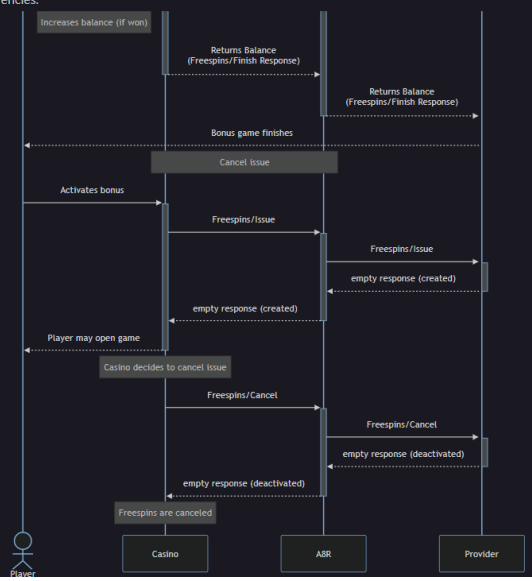
1. `bet_levels` is a set of rules (restrictions) for configuring free spins with regard to different currencies.



## The dependency of free spin issuance on bet levels

1. `bet_levels` is a set of rules (restrictions) for configuring free spins with regard to different currencies.

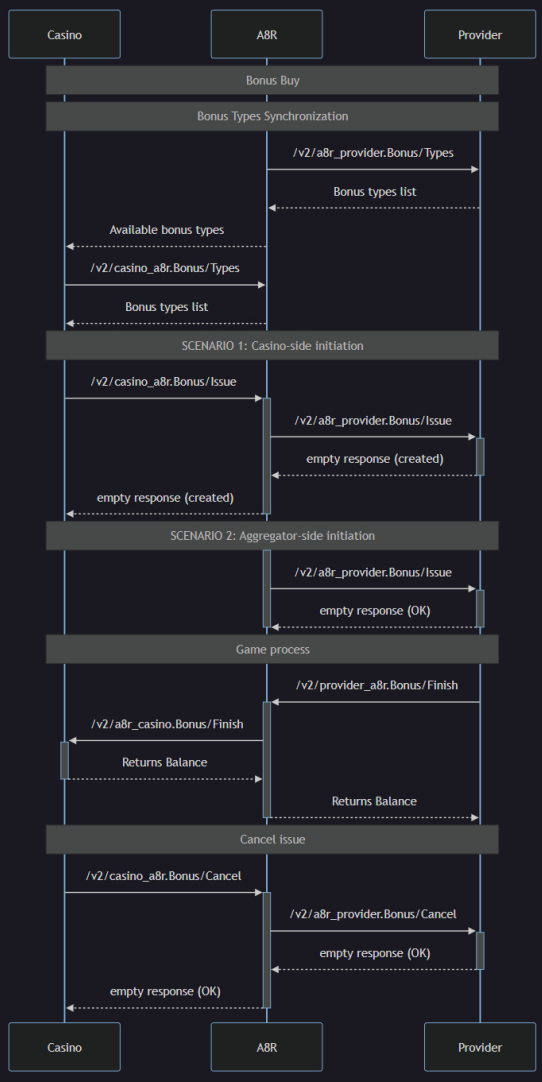
### The dependency of free spin issuance on bet levels

1. `bet_levels` is a set of rules (restrictions) for configuring free spins with regard to different currencies.
1. The Bonus Buy round is handled entirely on the Provider's side during the game session.
2. The Provider applies Bonus Buy rules and calculates the purchase cost. The `cost_multiplier` indicates how many times the Player must multiply their current bet to determine the activation cost of Bonus Buy in a specific game.
3. After the purchased bonus round is completed, the Provider calculates the total winnings.
4. The Provider sends `/v2/provider_a8r.Bonus/Finish` request to A8R (which then passes `/v2/a8r_casino.Bonus/Finish` request to the Casino) with the total amount of winnings.
5. The Casino increases the Player's balance (if applicable) and returns the updated balance in the response.
6. A8R forwards the updated balance back to the Provider and the game flow is completed.

### Bonus Buy Cancelation

The Casino can cancel the Bonus Buy if it hasn't expired yet and if deactivation is allowed according to Provider rules. This can be done by sending a `/v2/casino_a8r.Bonus/Cancel` request, which A8R forwards to the Provider via `/v2/a8r_provider.Bonus/Cancel`.

Find the more detailed workflow diagram below:




## How to launch games

This guide provides instructions on how to launch a game.

### Prerequisites

The game launch URL received from the server can be in one of two formats:

- A standard HTTPS URL.
- A data:URL (a base64-encoded string).

If your platform's game launcher does not support processing the data:URL format, you can request our connector.js script. Please contact our Game Aggregator Manager to obtain it and for detailed integration instructions.

## Steps

Handling of the launch URL can be implemented using one of two scenarios. Choose the one that best fits your technical capabilities.

**Scenario 1: Launching via URL (Standard Game Launcher)**

1. Obtain the `'<server response>'` with the game URL and session properties.
   - For a game session without real money, use the **POST** `/v2/casino_a8r.Launcher/Demo` endpoint.
   - For a game session with real money, use the **POST** `/v2/casino_a8r.Launcher/Real` endpoint.
2. Extract the game URL from the server response.
3. Using your standard game launcher, load the received URL (HTTPS or data:URL) into the game container on your page.

**Scenario 2: Launching via Our Connector**

If you are using our connector.js script, our Game Aggregator Manager will provide you with all necessary technical details and integration steps.

## Promo flow

The promo feature consists of a set of actions that offer payment operations to players that are not related to a
specific game round.

**Promo debit**

This action involves debiting the player's balance for non-gameplay updates, such as tips.

**Promo win**

With this action, the player's balance is credited campaign prizes, such as tournaments or promotions.

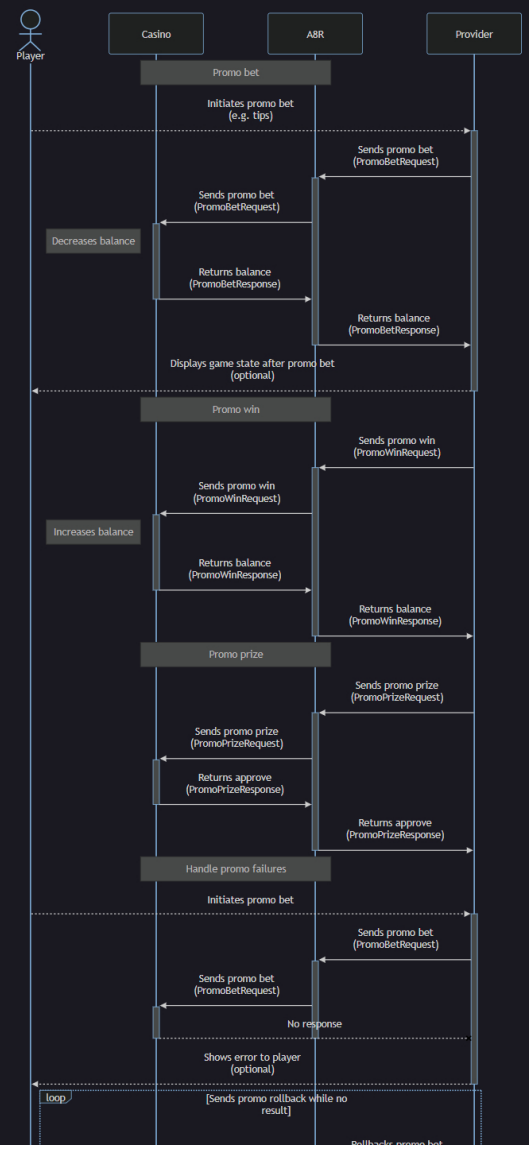
**Promo prize**

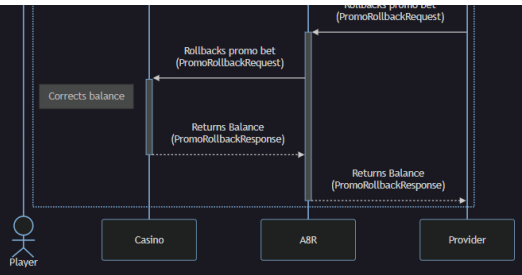
This action describes a non-cash prize that a player receives as a result of a non-game activity.

**Promo rollback**

If the original promo bet transaction was unsuccessful, this action involves refunding the original amount.

Find the more detailed workflow diagram below:






## FAQ

**Is it possible to transfer player's funds to Game Aggregator so Game Aggregator can process bets, wins, etc. and get funds back after game session finished?**

No, we use seamless wallet.

**What is the difference between `player_id` and `account_id` parameters?**

Game Aggregator uses `player_id` in communication with Casino. Every player has a single identifier among all player's accounts (wallets, currencies) and it is `player_id`.

In communication with Provider, Game Aggregator uses account_id which is an alias to combination of `player_id`, currency and `wallet_id`.

It means that `player_id` and `account_id` has one-to-many relationship.

**Do you have a bonus balance in your integration?**

Unfortunately, not. In our integration, we implement only balance in real money. But every Round/BetWin response contains `bonus_amount` parameter that shows what part of bet or win amount is concerned as bonus.

**Is that possible to have "bet" or "win" and "rollback" transactions in 1 request?**

It's not possible to send a combination of "bet" and "rollback" or "win" and "rollback" in one request as a "rollback" is another type of request aimed to cancel a "bet" or a "win". A "win" and a "bet" can be sent together.

**How many transaction types may one Round/BetWin request contain?**

A single request may contain 1 or more transactions. You can receive a "bet" transaction in one request, and a "win" transaction in the next request. Or you can receive both a "bet" and a "win" transactions in a single request.

**Is it possible to refer a win to a specific bet?**

A win can not always be matched with a specific bet (e.g. blackjack's "double"). Bets and wins can be associated only with the same `round_id` (game round). There also can be several bets and no win within one `round_id`.

**How do we use the information from "Jackpot Feed"?**

You can receive the current amount of jackpot for a game through the URL given in the documentation: it's just the way to let players know what is the possible prize in the jackpot game. But we can only pass this information if the game Provider has this feature integrated. Some providers have different levels of jackpots (from higher to lower) and this information can also be fetched.

**Is the Provider always set to 01.tech AggregatorS?**

At the test environment only 01.tech Aggregator games are available. When you pass all the tests and switch to the production, you will be able to have more Providers.

**What is the expected value for amount parameter if during freespins bonus play nothing has been won?**

"Amount" is a mandatory parameter. In case of nothing has been won `amount == "0.00"` is expected.

**Do you need to whitelist any IP on your side?**

We do not whitelist IPs on our side, we only use hash-based message authentication (see Security).

**Can you explain regarding Free Spin feature?**

Freespin feature is a promotional tool, which allows you to issue free games bonuses for your players.

When you send us a request to issue a freespin bonus for a player for a game (Freespins/Issue), it means, that the next time this player launches the game, he will start a bonus round. When to send this "issue freespins" request is up to you.

For example, in our platform it takes 2 steps: the operator prepares a template for the bonus in the back office (stating the Provider, the game or several games and the currency), issues this bonus for a specific player => the player gets a notification about the bonus and presses a button to activate it. When the button is pressed, the platform sends to Game Aggregator a Freespins/Issue request. Then the player can launch the game and play the issued number of spins for free.

**Do we have to create session for the player when issuing freespins?**

No, these are two unrelated requests. `round_id` identifiers are used in real money play, `issue_id` is used in freespins bonus rounds.

**How do we count freespins being played?**

There is no need to count the number of freespins: when the whole bonus is played, you receive a request with parameters `issue_id` (for you to identify the bonus) and `amount` (total winning in the bonus round). Freespins are not included in GGR.

**Is the player's bonus balance included in GGR?**

Since we do not distinguish between "real" and "bonus" balances, bonus bet or winning is counted as a

regular bet or winning and is a part of GGR.

The tip bet is a special type of debit that is not a part of the game financial transactions.

## API Protocol

Game Aggregator API v2 uses the Twirp Wire Protocol (v7) as the main protocol for communication between all parties. The main advantages of this protocol are:

1. API contracts are described in proto files, making API versioning easy and safe. In case of changes (such as adding new fields or endpoints), the old clients and servers will still work.
   - Code generators are used to produce server/client implementations in different languages.
   - API documentation is generated from the proto files.
   - Proto extensions such as validation and documentation can be used.
   - It supports both application/json and application/protobuf as content types and uses HTTP/1.
2. Any HTTP client can be used to interact with the API, such as curl, Postman, or Swagger, or a generated Twirp client can be used. Server and client implementations can be created in any language using native language functionality.

> TIP
>
> Game Aggregator will handle requests with both JSON and protobuf content types (responding in the type of the request), but requests made by Game Aggregator to the Casino and Provider will always be in JSON.

## Security

- Each request between servers must be signed.
- Each request signature must be validated.

The signature of the request shall be calculated using the HMAC-SHA256 algorithm, where the message is the request body and the key is "AUTH_TOKEN".

The signature shall be sent in the "**X-REQUEST-SIGN**" header of the request.

In case of a signature mismatch, the server shall respond with a Forbidden error (see Errors format).

```
// 400 Bad Request

{
  "code": "invalid_argument",
  "msg": "Forbidden.",
  "meta": {
    "api_code": "403",
    "api_message": "Forbidden."
  }
}
```

Examples of implementing signing in different languages:

### Generating signature in Go

```
package example

import (
    "crypto/hmac"
    "crypto/sha256"
    "encoding/hex"
)

func Sign(body, authToken []byte) (string, error) {
    mac := hmac.New(sha256.New, authToken)
    if _, err := mac.Write(body); err != nil {
        return "", err
    }

    return hex.EncodeToString(mac.Sum(nil)), nil
}
```

### Generating signature in Ruby

```
require 'openssl'

REQUEST_BODY = '{"user_id": "1196", "currency": "EUR", "game": "SuperSlot"}'
AUTH_TOKEN = '123456'

puts OpenSSL::HMAC.hexdigest(OpenSSL::Digest.new('sha256'), AUTH_TOKEN, REQUEST_BODY)
```

### Generating signature in Java

```
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import java.nio.charset.StandardCharsets;
import java.security.GeneralSecurityException;
import java.util.HexFormat;

class Main {
    private static final HexFormat formatter = HexFormat.of();

    public static String calculateHmacSHA256(byte[] data, int count, String secretKey) throws
GeneralSecurityException {
        Mac sha256Hmac = Mac.getInstance("HmacSHA256");
        SecretKeySpec secretKeySpec = new
SecretKeySpec(secretKey.getBytes(StandardCharsets.UTF_8), "HmacSHA256");
        sha256Hmac.init(secretKeySpec);
        sha256Hmac.update(data, 0, count);
        byte[] macData = sha256Hmac.doFinal();
        return formatter.formatHex(macData);
    }

    public static void main(String[] args) throws GeneralSecurityException {
        String AUTH_TOKEN = "12345";
        String REQUEST_BODY = "{\"user_id\": \"1196\", \"currency\": \"EUR\", \"game\":
\"SuperSlot\"}";
```

```
        byte[] bodyData = REQUEST_BODY.getBytes(StandardCharsets.UTF_8);
        String signature = calculateHmacSHA256(bodyData, bodyData.length, AUTH_TOKEN);

        System.out.println(signature);
    }
}
```

### Generating signature in JS

```
import crypto from "crypto";

const AUTH_TOKEN = "12345";
const REQUEST_BODY = `{"user_id": "1196", "currency": "EUR", "game": "SuperSlot"}`;

const signature = crypto
    .createHmac('sha256', AUTH_TOKEN)
    .update(REQUEST_BODY)
    .digest('hex')

console.log(signature);
```

### Generating signature in Postman

```
const AUTH_TOKEN = '123456';

const signature = CryptoJS.HmacSHA256(pm.request.body.toString(), authToken).toString();

pm.request.headers.add({key: "X-REQUEST-SIGN", value: hash});
```

## Data format

### Money Amount Format

Money Amount is passed as a String: The integer part of a number can have up to 18 digits and the fractional part can have up to 12 digits. The two parts are separated by a decimal point ("."). All amounts must be above or equal zero.

Money Amount Format Example:

```
"12.9999", "100", "1", "0", "0.0000003"
```

```
^-?\d{1,18}(\.\d{1,12})?$
```

### Currency format

Game Aggregator works with two different types of currencies:

1. Fiat currencies described on ISO_4217 with a 3-letter identifier in uppercase.
2. Crypto currencies with a 2 or 15-letter identifier in uppercase.
3. Denominated crypto currencies are not supported.

```
"EUR", "USD", "USDT", "DASH", "BTC", "DOG"
```

```
^[A-Z0-9]{2,15}$
```

### Country format

Use a 2-letter identifier according to ISO-3166-1 alpha-2 codes in uppercase for country identifiers.

```
"US", "EN", "DE"
```

```
^[A-Z]{2}$
```

### Jurisdiction format

Jurisdiction is an identifier of the license. Use a 2-letter identifier of a country (see Country format) or a combination of 2-letter identifiers if applying to country's region ("CA-ON" for Ontario, Canada) in uppercase.

```
"US", "DE", "CW", "CA-ON"
```

```
^[A-Z]{2}(-[A-Z]{2})?$
```

### Locale format

Use a 2-letter identifier according to ISO 639-1 in lowercase for the language identifier (locale). If an unsupported value is specified, the default language will be English - "en".

```
"us", "en", "ar"
```

```
^[a-z]{2}$
```

### Time Format

Date and time shall be formatted according to RFC 3339 and sent only in UTC. The only valid datetime format is `1985-12-31T23:20:50[.999999999]Z`, where:

- "1985-12-31T23:20:50" is the mandatory part.
- [.999999999] is the optional nanoseconds part and may have from 1 to 9 digits after the period (e.g. ".5" == ".500000000").

- [Z] is the mandatory timezone offset. "Z" represents UTC/GMT offset and must always be "Z".

Time Format examples:

```
1985-12-31T00:00:00Z (for date).
1985-12-31T23:20:50Z (for date with time).
1985-12-31T23:20:50.771238663Z (for date with time and nanoseconds).
```

```
^\d{4}-\d{2}-\d{2}T\d{2}:\d{2}:\d{2}(\.\d{1,9})?Z$
```

### URL format

URL shall be formatted according to RFC 3986

```
"https://www.example.com/index.html"
```

```
https?:\\/\\/([^\\/?#]+)([^?#]*)\\\?[^#]*)?(#.*)?$
```

## Errors Format

Standard Twirp error structure and error codes are described in Twirp documentation.

Main points to know about Twirp errors:

1. Errors are always returned with Content-Type `application/json` and never `application/protobuf`.
2. Error response will look like this:

```
// 400 Bad Request

{
  "code": "invalid_argument",
  "msg": "Business error",
  "meta": {
    "api_code": "110",
    "api_message": "Player is disabled",
    "balance": "0.01"
  }
}
```

We use only two **twirp** and **HTTP** error codes to simplify API behavior:

1. HTTP code `400 Bad Request` - Twirp code `invalid_argument`
2. HTTP code `500 Internal Server Error` - Twirp code `internal`

The actual error code is passed through `meta.api_code` field.

Possible options in the "meta" field:

| Parameter | Type | Mandatory | Description |
|---|---|---|---|
| api_code | string | Required | Error code (**not** HTTP code). |
| api_message | string | Required | Error description. |
| balance | string | Optional | Player's balance. If known. Can be added only in responses from Casino to Provider (mandatory if api_code: 100, 105, 106). |

### Error Examples

```
// 400 Bad Request

{
  "code": "invalid_argument",
  "msg": "Game: value length must be at least 1 symbol",
  "meta": {
    "api_code": "400",
    "api_message": "Game: value length must be at least 1 symbols"
  }
}
```

```
// 400 Bad Request

{
  "code": "invalid_argument",
  "msg": "Not enough funds.",
  "meta": {
    "api_code": "100",
    "api_message": "Not enough funds.",
    "balance": "0.01"
  }
}
```

### Available error API codes

| api_code | api_message | comment |
|---|---|---|
| 100 | Player has not enough funds to process an action. | |
| 101 | Player is invalid or not found | |
| 105 | Player reached customized bet limit. | |
| 106 | Bet exceeded max bet limit. | |
| 107 | Game is forbidden to the player. | |
| 110 | Player is disabled. | |
| 125 | Game does not support demo mode. | |

| | | |
|---|---|---|
| 153 | Game is not available in Player's country. | |
| 154 | Currency is not allowed for the player. | |
| 155 | Forbidden to change already set field. | When user's email and currency already set and you provide different email or currency you will get this error. |
| 400 | Bad request. | Badly formatted JSON. |
| 403 | Forbidden. | Request sign doesn't match. |
| 404 | Not found. | |
| 405 | Game is not available to your casino. | |
| 406 | Freespins are not available for your casino. | |
| 409 | Already exists. | |
| 410 | Games have different customers. | |
| 413 | Unknown session. | |
| 500 | Unknown error. | |
| 502 | Unknown error in external service. | |
| 503 | Service is unavailable. | |
| 504 | Request timed out. | |
| 600 | Game provider doesn't provide freespins. | |
| 601 | Impossible to issue freespins in requested game. | |
| 602 | You shall provide at least one game to issue freespins. | |
| 603 | Bad expiration date. | Expiration date shall be in future and freespins shall not be active for more than 1 month. |
| 605 | Can't change issue state from its current to requested. | |
| 606 | You can't change issue state when issue status is not synced. | |
| 607 | Can't issue freespins for different game providers. | |
| 610 | Invalid freespins issue. | |
| 611 | Freespins issue has already expired. | |
| 620 | Freespins issue can't be canceled. | |
| 700 | Requested live game is not available right now. | |

## Gamelist Format

Gamelist contains an array of game items. The data is provided in YAML format. Each game item has the following fields:

| Parameter | Type | Example value | Mandatory | Description |
|---|---|---|---|---|
| title | string | `Platinum Lightning` | Required | Game name (W specifying RTP) |
| id | string | `PlatinumLightning` | Required | Unique identifi game by which Aggregator an casino identify Casinos must u parameter, for to launch a gar |
| ref | string | `01tech:PlatinumLightning` | Optional | Game referenc `provider:id` Returned only enabled for yo Disabled by de |
| id2 | string | `6911806e-6a26-43be-81fa-00754b8c1099` | Required | Unique identifi game by which Aggregator an provider identi game. |
| provider | string | `01tech` | Required | Unique identifi game provider. provider can in multiple produ Please use a cle |

| | | | | | |
|---|---|---|---|---|---|
| | | | | | alphabetic valu |
| | producer | string | `bgaming` | Optional | Unique identifi game produce use a clear alpl value. It is nece specify the exa the game prod |
| | category | string | `slots` | Mandatory | Game's catego restricted to a set of values; n categories may introduced at a Possible values<br>• card<br>• casual<br>• crash<br>• craps<br>• fishing<br>• lottery<br>• mines<br>• poker<br>• roulette<br>• scratch<br>• slots<br>• video_pok<br>• virtual_spo |
| | theme | string | `sweets` | Optional | Game theme. T value is other. values:<br>• africa<br>• ancient_ci<br>• asia<br>• book_off<br>• branded<br>• easter<br>• egypt<br>• fantasy<br>• food<br>• fruits<br>• girls<br>• halloween<br>• horrors<br>• joker<br>• luxury_life<br>• military<br>• other<br>• party<br>• pirates<br>• retro<br>• space<br>• sport<br>• st_patrick_<br>• st_valentin<br>• sweets<br>• underwate<br>• western<br>• x_mas_and |
| | has_freespins | bool | `true` | Optional | If true, then the supports issuar spins via the AI false otherwise |
| | feature_group | string | `basic` | Required | Financial group game based or conditions. |
| | customised | bool | `false` | Required | If true, then the customised for particular casin otherwise.Note customized gar designed speci a particular cas can only be use casino. |
| | devices | array[string] | `["mobile"]` | Required | Array of strings each string is a supported devi game. Possible<br>• mobile<br>• desktop |
| | | | | | Array of strings |

| | | | | |
|---|---|---|---|---|
| licenses | array[string] | `["MT"]` | Required | each string is a that the game certified with. licenses format |
| jackpot_type | string | `Not Available` | Required | Indicates the ty jackpot campai game participa Possible values<br>• Not Availa Jackpot ty specified. default.<br>• Network: J games tha provided f of casinos.<br>• Local: Jack games tha provided c one casino<br>• In game: G integrated in-game ja feature. |
| forbid_bonus_play | bool | `true` | Required | If true, then pla use bonus func the game; false otherwise. |
| lines | int | `25` | Optional | Number of line game. For slot only. |
| ways | int | `243` | Optional | Number of win combinations. games only. |
| description | string | `Cherry Fiesta is a fruit-themed video slot game with 5 reels and 9 paylines which involves 13 different symbols.` | Optional | Game descripti |
| has_live | bool | `true` | Required | If true, then the streamed in rea with real dealer shuffling, dealir interacting with players; false ot |
| payout | string | `98` | Optional | Percentage valu shows how mu will be returned player from eac they make. For games only. |
| hit_rate | string | `38.8` | Optional | Frequency of w 100 bets. The h hit rate the low volatility rating versa. For slot only. |
| volatility_rating | string | `medium-high` | Optional | Winning expec the game. This shows how ofte how much a pl expect to win d their playing se the volatility ra low, the player bets in the gam larger amount versa. For slot only. |
| has_jackpot | bool | `true` | Optional | If true, then the supports the ja feature; false ot |
| hd | bool | `true` | Optional | If true, then the supports wides format; false ot |
| released_at | timestamp | `2018-06-05` | Optional | Game release c can't launch th before this date |
| recalled_at | timestamp | `2018-02-13` | Optional | Game recall da can't launch th after this date. |
| bonus_buy | bool | `true` | Required | If true, then pla buy bonuses to the game; false |

| | | | | |
|---|---|---|---|---|
| | | | | otherwise. |
| multiplier | int | `2000` | Optional | Maximum coef<br>calculating the<br>For slot games |
| restrictions | object | `default:`<br>`  blacklist:`<br>`  - AF`<br>`  - AU` | Optional | Contains inform<br>restricted black<br>allowed whiteli<br>countries for a<br>countries forma |

Modified at 9 days ago

Next
Aggregator API for Provider →