

Games Aggregator Guide

Contents

About this document	3
Purpose	3
Intended Audience	3
API	4
Game Aggregator – Universal API	4
API Conventions	4
Security	4
Common handling logic for all API Requests.....	5
initGame	5
debit.....	7
credit.....	8
debitAndCredit	11
balance	15
rollbackDebit.....	16
verifyPlayer.....	18
Game aggregator - Launch game API	19
Launching a game	19
Game History	21
Game aggregator – Checksum tool API	22
Get Checksums.....	22
Get Checksums By Game ID	24
Game aggregator – Free Rounds API.....	25
Issue FreeRounds.....	25
Cancel FreeRounds	27
Regulations	28
UKGC - RTS 13	29
Appendix B - Game Aggregator error handling.....	30
Appendix C - API additionalData JSON	31

About this document

Purpose

Anakatech Aggregator Guide is a simplified API mechanism for all Anaktech & 3rd party provided games.

Anakatech's game aggregator allows its game providers to integrate via a single robust API and receive gaming services from a variety of compatible game providers

The standardised API calls for all game types and game providers while minimizing the number of parameters needed for game's operation.

The game aggregator provides a single point of integration and will take care of the integration complexity and requirements for each of the game providers while keeping a single universal API to its game providers.

The game aggregator will provide other capabilities such as game history and backoffice reporting

Intended Audience

Technical teams of the game providers

API

Game Aggregator – Universal API

The Game aggregator requires the game providers to integrate using the universal API. Once the integration takes place, the game provider will be able to provide your games through the game aggregator.

The universal API contains the following methods:

- 1) **initGame**
- 2) **debit**
- 3) **credit**
- 4) **debitAndCredit**
- 5) **balance**
- 6) **rollbackDebit**
- 7) **verifyPlayer**

API Conventions

This document API definitions are using the following conventions:

- Parameter name - the name of the parameter as it should appear in the request.
- Parameter type - describes the acceptable value of the parameter.
 - a - alphabet
 - n - numeric
 - an - alphanumeric
 - an..10 - alphanumeric up to 10 characters in length
 - an 10 - alphanumeric with length of exactly 10 characters.
 - c - collection of values

Required - can be one of: Mandatory, Optional or Conditional. The latest means that in certain conditions the parameter becomes mandatory.

Security

All communication between the Game aggregator and the game providers are using HTTPS. The game provider must verify that calls made to it originate from the Game aggregator. This is done by:

Checking the IP Address against the whitelisted Game Aggregator Addresses. Whitelisted IP Addresses are available through the Game Aggregator Backoffice.

Shared Secret - The game provider needs to define a shared secret using the Game aggregator backoffice. The Game aggregator will include the shared secret in every call it makes. The game provider must secret the shared secret before allowing any calls to be processed.

Common handling logic for all API Requests

initGame

The initGame method is called right after the player is launching the game. This represents the beginning of a new session within this game for this player.

Function Signature

Https://game_aggregator_url/initGame

HTTP Method

POST

Input

Parameter name	Parameter Type	Restrictions	Required	Parameter description	Remarks
secret	String	an..24	Mandatory	This is the shared secret between the game aggregator and the game provider	The secret is configured by the Game aggregator admin for each separately. See Security section above.
sessionId	String	an..256	Mandatory	The unique session id for the new session.	Each game launching will create a new session.
securityToken	String	an..1024	Mandatory	The security token is provided so that the wallet platform can uniquely identify the player's session	The security token was previously provided to the Game aggregator upon launching the game.
playerId	String	an..32	Mandatory	The ID of the player's account in the game provider's system	Uniquely identifies a player account per brand.
playMode	int	n 1	Mandatory	The play mode which can be one of: 1 – real money 3 – anonymous	
gameProviderId	int	n..32	Mandatory	The unique ID of the provider	Will be provided to you by your account manager
providerGameId	String	an..256	Mandatory	The game ID	Each game has a unique ID.

Example:

```
{
  "secret" : "ab8sj330jckw8w1yc",
  "sessionId" : "1234-abcde-5678-zxcv",
  "securityToken" : "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9",
  "playerId" : "SM339072",
  "playMode" : 1,
  "gameProviderId" : 44,
  "providerGameId" : "game12"
}
```

Output

Parameter name	Parameter Type	Restrictions	Parameter description	Remarks
success	Boolean		true in case the request succeeded, false otherwise.	See Appendix B for detailed output structure in case of errors.
balance	long	n..32		
cashBalance	long	n..32	The player's account cash balance in cents	Used in regulated markets
bonusBalance	long	n..32	The player's account bonus balance in cents	Used in regulated markets
additionalData	Object	a...512	Provided only when free rounds \ jackpot data are available on the game provider's platform. Also includes the <i>maxTotalBetAmount</i> which player is allowed to be on	See Appendix C for more details

Example:

```
{
  "success":true,
  "result":false,
  "balance":36961,
  "cashBalance":36961,
  "bonusBalance":0,
  "additionalData":
    {"freeRoundsBet":null,
     "totalFreeRounds":null,
     "playedFreeRounds":null,
     "freeRoundsWinnings":null,
     "jackpotGameRoundDetails":null,
     "maxTotalBetGameCategory":"VIDEO_SLOT",
     "maxTotalBetAmount":500.0}
}
```

debit

The debit method is called when the player makes a bet.

Function Signature

Https://game_aggregator_url/debit

HTTP Method

POST

Input

Parameter name	Parameter Type	Restrictions	Required	Parameter description	Remarks
secret	String	an..24	Mandatory	This is the shared secret between the game aggregator and the game provider	The secret is configured by the Game aggregator admin for each game provider separately. See Security section above.
sessionId	String	an..256	Mandatory	The unique session id for the new session.	Each game launching will create a new session.
securityToken	String	an..1024	Mandatory	The security token is provided so that the wallet platform can uniquely identify the player's session	The security token was previously provided to the Game aggregator upon launching the game.
playerId	String	an..32	Mandatory	The ID of the player's account in the game provider's system	Uniquely identifies a player account per brand.
playMode	int	n 1	Mandatory	The play mode which can be one of: 1 – real money 3 – anonymous	
gameProviderId	int	n..32	Mandatory	The unique ID of the provider	Will be provided to you by your account manager
providerGameId	String	an..256	Mandatory	The game ID	Each game has a unique ID.
roundId	String	an..256	Mandatory	The unique round ID	
secondaryRoundId	String	an..256	Optional	The secondary round identifier which was provided for the original round (used for games free round / bonus round feature \ multiple credit\debit)	
transactionId	String	a..36	Mandatory	The unique transaction ID of the debit	
currency	String	an 3	Mandatory	The ISO 4217 code	e.g. EUR
amount	long	n..32	Mandatory	The bet amount made in cents	Can be only positive.
closeRound	Boolean				closeRound = true means no further request for the given roundId should be handled.
additionalData	Object	a...512	Optional	Provided only when free rounds \ jackpot	See Appendix C for more details

				data are available on the game provider's platform	
--	--	--	--	--	--

Example:

```
{
  "secret" : "ab8sj330jckw8w1yc",
  "sessionId" : "1234-abcde-5678-zxcv",
  "securityToken" : "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9",
  "playerId" : "SM339072",
  "playMode" : 1,
  "gameProviderId" : 44,
  "providerGameId" : "game12",
  "roundId" : "123:1234-abcde-5678-zxcv",
  "transactionId" : "d:1234-zzzz-5678-zzzz",
  "currency" : "EUR",
  "amount" : 10
}
```

Output

Parameter name	Parameter Type	Restrictions	Parameter description	Remarks
success	Boolean		true in case the request succeeded, false otherwise.	See Appendix B for detailed output structure in case of errors.
referenceId	String	an..256	Unique ID of the record in the game provider's system.	This ID can later be used for both troubleshooting and reports, in order to reference the data as it was recorded in the game provider's system.
balance	long	n..32	The player's account balance in cents after the request was handled	
cashBalance	long	n..32	The player's account cash balance in cents after the request was handled	Used in regulated markets
bonusBalance	long	n..32	The player's account bonus balance in cents after the request was handled	Used in regulated markets
currency	String	an 3	The ISO 4217 code	e.g. EUR
additionalData	Object	a...512	Provided only when free rounds \ jackpot data are available on the game provider's platform	See Appendix C for more details

credit

The credit method is called when the player balance needs to be added with the credit amount (win). This request can mark the end of a round by indicating the parameter 'closeRound' = TRUE.

Function Signature

Https://game_aggregator_url/credit

HTTP Method

POST

Input

Parameter name	Parameter Type	Restrictions	Required	Parameter description	Remarks
secret	String	an..24	Mandatory	This is the shared secret between the game aggregator and the game provider	The secret is configured by the Game aggregator admin for each game provider separately. See Security section above.
sessionId	String	an..256	Mandatory	The unique session id for the new session.	Each game launching will create a new session.
securityToken	String	an..1024	Mandatory	The security token is provided so that the wallet platform can uniquely identify the player's session	The security token was previously provided to the Game aggregator upon launching the game.
playerId	String	an..32	Mandatory	The ID of the player's account in the game provider's system	Uniquely identifies a player account per brand.
playMode	int	n 1	Mandatory	The play mode which can be one of: 1 – real money 3 – anonymous	
gameProviderId	int	n..32	Mandatory	The unique ID of the provider	Will be provided to you by your account manager
providerGameId	String	an..256	Mandatory	The game ID	Each game has a unique ID.
roundId	String	an..256	Mandatory	The unique round ID	
secondaryRoundId	String	an..256	Optional	The secondary round identifier which was provided for the original round (used for games free round / bonus round feature \ multiple credit(debit)	
transactionId	String	a..36	Mandatory	The unique transaction ID of the credit	
currency	String	an 3	Mandatory	The ISO 4217 code	e.g. EUR
amount	long	n..32	Mandatory	The win amount made in cents	Can be 0 or positive.
closeRound	Boolean				closeRound = true means no further request for the given roundId should be handled.
additionalData	Object	a...512	Optional	Provided only when free rounds \ jackpot data are available on the game provider's platform	See Appendix C for more details

Example:

```
{
  "secret" : "ab8sj330jckw8w1yc",
  "sessionId" : "1234-abcde-5678-zxcv",
  "securityToken" : "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9",
  "playerId" : "SM339072",
  "playMode" : 1,
  "providerGameId" : "game12",
  "gameProviderId" : 44,
  "roundId" : "123:1234-abcde-5678-zxcv",
  "transactionId" : "c:1234-zzzz-5678-zzzz",
  "currency" : "EUR",
  "amount" : 0,
  "closeRound": true
}
```

Output

Parameter name	Parameter Type	Restrictions	Parameter description	Remarks
success	Boolean		true in case the request succeeded, false otherwise.	See Appendix B for detailed output structure in case of errors.
referenceId	String	an..256	Unique ID of the record in the game provider's system.	This ID can later be used for both troubleshooting and reports, in order to reference the data as it was recorded in the game provider's system.
balance	long	n..32	The player's account balance in cents after the request was handled	
cashBalance	long	n..32	The player's account cash balance in cents after the request was handled	Used in regulated markets
bonusBalance	long	n..32	The player's account bonus balance in cents after the request was handled	Used in regulated markets
currency	String	an 3	The ISO 4217 code	e.g. EUR
additionalData	Object	a...512	Provided only when free rounds \ jackpot data are available on the game provider's platform	See Appendix C for more details

Example:

```
{
  "success" : true,
  "referenceId" : "TR66382",
  "balance" : 200,
  "cashBalance" : 50,
  "bonusBalance":150,
  "currency": "EUR"
}
```

}

debitAndCredit

The debitAndCredit method is an efficient way to communicate both multiple credit transactions and multiple debit transactions in a single request. This method is used with compatible game providers only. It's important that all credit\debit transactions are atomic - either all of them were committed or none of them were.

Function Signature

Https://game_aggregator_url/debitAndCredit

HTTP Method

POST

Input

Parameter name	Parameter Type	Restrictions	Required	Parameter description	Remarks
secret	String	an..24	Mandatory	This is the shared secret between the game aggregator and the game provider	The secret is configured by the Game aggregator admin for each game provider separately. See Security section above.
sessionId	String	an..256	Mandatory	The unique session id for the new session.	Each game launching will create a new session.
securityToken	String	an..1024	Mandatory	The security token is provided so that the wallet platform can uniquely identify the player's session	The security token was previously provided to the Game aggregator upon launching the game.
playerId	String	an..32	Mandatory	The ID of the player's account in the game provider's system	Uniquely identifies a player account per brand.
playMode	int	n 1	Mandatory	The play mode which can be one of: 1 – real money 3 – anonymous	
gameProviderId	int	n..32	Mandatory	The unique ID of the provider	Will be provided to you by your account manager
providerGameId	String	an..256	Mandatory	The game ID	Each game has a unique ID.
roundId	String	an..256	Mandatory	The unique round ID	
transactions	Array of Transaction object	array	Mandatory	Each object in the array is described below	
closeRound	Boolean				closeRound = true means no further request for the given roundId should be handled. Usually debitAndCredit requests are having closeRound = true but not necessarily
additionalData	Object	a...512	Optional	Provided only when free rounds \ jackpot	See Appendix C for more details

				data are available on the game provider's platform	
--	--	--	--	--	--

Transaction (object)

Parameter name	Parameter Type	Restrictions	Required	Parameter description	Remarks
type	String	an..5	Mandatory	The type of the transaction: CREDIT \ DEBIT	
transactionId	String	a..36	Mandatory	The unique transaction ID	
secondaryRoundId	String	an..256	Optional	The secondary round identifier which was provided for the original round	
currency	String	an 3	Mandatory	The ISO 4217 code	e.g. EUR
amount	long	n..32	Mandatory	The amount in cents	For type = CREDIT can be 0 or positive. For type = DEBIT can be 0 or positive.

Example:

```
{
  "secret" : "ab8sj330jckw8w1yc",
  "sessionId" : "1234-abcde-5678-zxcv",
  "securityToken" : "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9",
  "playerId" : "SM339072",
  "playMode" : 1,
  "gameProviderId" : 44,
  "providerGameId" : "game12",
  "roundId" : "123:1234-abcde-5678-zxcv",
  "closeRound": true,
  "transactions": [
    {
      "type" : "DEBIT",
      "transactionId" : "d:1234-zzzz-5678-zzzz",
      "currency" : "EUR",
      "amount" : 10
    },
    {
      "type" : "CREDIT",
      "transactionId" : "c:6546-zzzz-5678-mmmm",
      "currency" : "EUR",
      "amount" : 10
    }
  ]
}
```


Output

Parameter name	Parameter Type	Restrictions	Parameter description	Remarks
success	Boolean		true in case the request succeeded, false otherwise.	See Appendix B for detailed output structure in case of errors.
referenceId	String	an..256	Unique ID of the record in the game provider's system.	This ID can later be used for both troubleshooting and reports, in order to reference the data as it was recorded in the game provider's system.
balance	long	n..32	The player's account balance in cents after the request was handled	
cashBalance	long	n..32	The player's account cash balance in cents after the request was handled	Used in regulated markets
bonusBalance	long	n..32	The player's account bonus balance in cents after the request was handled	Used in regulated markets
currency	String	an 3	The ISO 4217 code	e.g. EUR
additionalData	Object	a...512	Provided only when free rounds \ jackpot data are available on the game provider's platform	See Appendix C for more details

Example:

```
{
  "success" : true,
  "referenceId" : "TR66382",
  "balance" : 200,
  "cashBalance" : 50,
  "bonusBalance":150,
  "currency": "EUR"
}
```

balance

The balance() method is called whenever the player balance is required.

Function Signature

Https://game_aggregator_url/balance

HTTP Method

POST

Input

Parameter name	Parameter Type	Restrictions	Required	Parameter description	Remarks
secret	String	an..24	Mandatory	This is the shared secret between the game aggregator and the game provider	The secret is configured by the Game aggregator admin for each game provider separately. See Security section above.
sessionId	String	an..256	Mandatory	The unique session id	
securityToken	String	an..1024	Mandatory	The security token is provided so that the wallet platform can uniquely identify the player's session	The security token was previously provided to the Game aggregator upon launching the game.
playerId	String	an..32	Mandatory	The ID of the player's account in the game provider's system	Uniquely identifies a player account per brand.
playMode	int	n 1	Mandatory	The play mode which can be one of: 1 – real money 3 – anonymous	Real balance \ fun balance
gameProviderId	int	n..32	Mandatory	The unique ID of the provider	Will be provided to you by your account manager
providerGameId	String	an..256	Mandatory	The game ID	Each game has a unique ID.

Example:

```
{
  "secret" : "ab8sj330jckw8w1yc",
  "sessionId" : "1234-abcde-5678-zxcv",
  "securityToken" : "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9",
  "playerId" : "SM339072",
  "gameProviderId" : 44,
  "providerGameId" : "game12",
  "playMode" : 1
}
```

Output

Parameter name	Parameter Type	Restrictions	Parameter description	Remarks
success	Boolean		true in case the request succeeded, false otherwise.	See Appendix B for detailed output structure in case of errors.
balance	long	n..32	The player's account balance in cents	
cashBalance	long	n..32	The player's account cash balance in cents	Used in regulated markets
bonusBalance	long	n..32	The player's account bonus balance in cents	Used in regulated markets

Example:

```
{
  "success" : true,
  "balance" : 200,
  "cashBalance" : 50,
  "bonusBalance":150
}
```

rollbackDebit

The rollbackDebit method is called when a debit request is timed out causing an invalid state. This request will only follow a single 'debit' transaction, there is no rollback for debitAndCredit \ credit transactions. Rolledback rounds are not closed, future credit calls will make sure it is closed eventually.

Function Signature

https://game_aggregator_url/rollbackDebit

HTTP Method

POST

Input

Parameter name	Parameter Type	Restrictions	Required	Parameter description	Remarks
secret	String	an..24	Mandatory	This is the shared secret between the game aggregator and the game provider	The secret is configured by the Game aggregator admin for each game provider

					separately. See Security section above.
sessionId	String	an..256	Mandatory	The unique session id for the new session.	Each game launching will create a new session.
securityToken	String	an..1024	Mandatory	The security token is provided so that the wallet platform can uniquely identify the player's session	The security token was previously provided to the Game aggregator upon launching the game.
playerId	String	an..32	Mandatory	The ID of the player's account in the game provider's system	Uniquely identifies a player account per brand.
playMode	int	n 1	Mandatory	The play mode which can be one of: 1 – real money 3 – anonymous	
gameProviderId	int	n..32	Mandatory	The unique ID of the provider	Will be provided to you by your account manager
providerGameId	String	an..256	Mandatory	The game ID	Each game has a unique ID.
roundId	String	an..256	Mandatory	The unique round ID	
transactionId	String	a..36	Mandatory	The unique transaction ID of the debit we need to rollback	
currency	String	an 3	Mandatory	The ISO 4217 code	e.g. EUR
amount	long	n..32	Mandatory	The bet amount to be refunded	
closeRound	Boolean				closeRound = true means no further request for the given roundId should be handled. Usually debitAndCredit requests are having closeRound = true but not necessarily

Example:

```
{
  "secret" : "ab8sj330jckw8w1yc",
  "sessionId" : "1234-abcde-5678-zxcv",
  "securityToken" : "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9",
  "playerId" : "SM339072",
  "playMode" : 1,
  "gameProviderId" : 44,
  "providerGameId" : "game12",
  "roundId" : "123:1234-abcde-5678-zxcv",
  "transactionId" : "d:1234-zzzz-5678-zzzz",
  "currency" : "EUR",
  "amount" : 10,
  "closeRound": true
}
```

Output

Parameter name	Parameter Type	Restrictions	Parameter description	Remarks
success	Boolean		true in case the request succeeded, false otherwise.	See Appendix B for detailed output structure in case of errors.
referenceId	String	an..256	Unique ID of the record in the game provider's system.	This ID can later be used for both troubleshooting and reports, in order to reference the data as it was recorded in the game provider's system.
balance	long	n..32	The player's account balance in cents after the request was handled	
cashBalance	long	n..32	The player's account cash balance in cents after the request was handled	Used in regulated markets
bonusBalance	long	n..32	The player's account bonus balance in cents after the request was handled	Used in regulated markets
currency	String	an 3	The ISO 4217 code	e.g. EUR

Example:

```
{
  "success" : true,
  "referenceId" : "TR66382",
  "balance" : 200,
  "cashBalance" : 50,
  "bonusBalance":150,
  "currency": "EUR"
}
```

verifyPlayer

The verifyPlayer() method is called in case there is a need to verify a player's session. One use of this method is when presenting game history, and it is necessary to make sure we present game history only to the player who played these games.

Function Signature

Https://game_aggregator_url/verifyPlayer

HTTP Method

POST

Input

Parameter name	Parameter Type	Restrictions	Required	Parameter description	Remarks
secret	String	an..24	Mandatory	This is the shared secret between the game aggregator and the game provider	The secret is configured by the Game aggregator admin for each game provider

					separately. See Security section above.
securityToken	String	an..1024	Mandatory	The security token is provided so that the wallet platform can uniquely identify the player's session	The security token was previously provided to the Game aggregator upon launching the game.
playerId	String	an..32	Mandatory	The ID of the player's account in the game provider's system	Uniquely identifies a player account per brand.

Example:

```
{
  "secret" : "ab8sj330jckw8w1yc",
  "securityToken" : "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9",
  "playerId" : "SM339072"
}
```

Output

Parameter name	Parameter Type	Restrictions	Parameter description	Remarks
success	Boolean		true in case the request succeeded, false otherwise.	See Appendix B for detailed output structure in case of errors.
result	Boolean		true in case the security token is verified and belongs to the player, false otherwise.	

Example:

```
{
  "success" : true,
  "result" : true
}
```

Game aggregator - Launch game API

Launching a game

To launch a game, Game Aggregator sends a launchGame request with parameters to the game provider's endpoint. All parameters will be Base64URL encoded. The call will be redirected to the required game and the game will be launched by the game provider in an iframe.

Function Signature

Https://game_provider_url/launchGame

HTTP Method

GET

Input query parameters

Parameter Name	Parameter Type	Required	Description
customerId	an..128	Mandatory	The customer ID. Will be provided to you by your account manager.
brandId	an..128	Mandatory	The ID of the customer's brand. Will be provided to you by your account manager.
playMode	n 1	Mandatory	The play mode which can be one of: 1 – real money 3 – anonymous
securityToken	String..1024 Alphanumeric characters in addition to _ . - + = /	Mandatory	A token which will later be sent by the game server to the game provider's system to uniquely identify the player's session. Mandatory parameter except for anonymous play mode.
playerId	String..32 Alphanumeric characters in addition to @ _ - = () # % ! + * ' ,	Mandatory	The unique ID of the player's account in the game provider's system. Mandatory parameter except for anonymous play mode.
nickname	String..16 Alphanumeric characters in addition to @ _ - = () # % ! + * ' ,	Mandatory	The player's nickname. Mandatory parameter except for anonymous play mode
balance	n..10	Mandatory	The current player's balance in cents. In anonymous play mode the balance parameter will set the initial balance, and when the player tries to make a wager greater than his/her available balance, his/her balance will be reloaded automatically to the initial balance.
currency	a 3	Mandatory	ISO 4217 currency code. For example: "EUR", "GBP".
language	a 2	Mandatory	ISO 639-1 language code. For example: "en", "nl".
country	a 2	Mandatory	ISO 3166 country code. For example: "GB", "DE".
providerGameId	String..256	Mandatory	The unique game ID.
lobbyURL	String..256 A valid URL	Mandatory	URL to the game aggregator's lobby.
jurisdiction	a 2	Mandatory	ISO 3166 country code. Specifies the jurisdiction under which this game is being played. The game will enforce specific jurisdiction requirements based on this parameter.

			For example: use “GB” to conform with UKGC requirements. If not provided, no specific jurisdiction requirements are applied.
realityCheckInterval	n..4	Conditional	Specifies the time interval in seconds at which the reality check will appear on the screen. A zero value indicates no time interval which means that the reality check will never be presented (or presented just once if realityCheckStartTime is provided). Mandatory in case <i>jurisdiction</i> parameters are set to one which requires reality checks. If the specified jurisdiction does not require reality checks, or is not provided, but this parameter is provided, an error will be returned.
realityCheckStartTime	n..4	Optional	Indicates the seconds to wait until the 1st reality check should be presented. A zero value means that the reality check should be presented immediately w/o delay. If you don’t want to present a reality check at all, just set realityCheckInterval to zero, and don’t set realityCheckStartTime at all. This parameter provides a way to seamlessly continue previous reality checks presented during the player’s game session. If not provided, realityCheckInterval is used instead.
audio	n 1	Optional	The game audio mode which can be one of: 1 – on (default on PC) 2 – off 3 – ask player Audio mode 3 is only relevant for Mobile. On PC only modes 1 and 2 are valid. Player preferences made during the session will override this.
originUrl	String..256 A valid URL	Mandatory	The origin url of the lobby opening the game
minBet	n..11	Optional	The minimum bet allowed (for supported games only)
maxTotalBet	n..11	Optional	The max total bet allowed (for supported games only)
defaultBet	n..11	Optional	The default bet (for supported games only)

Game History

In order to get a game round history, the lobby will call GA that will construct the hereunder url for the lobby to be opened in an iframe. The game provider should implement this endpoint, that will be open only for the GA IP, to return a page with the requested round’s details – bets, wins, visual result or a link to replay video (if supported), time played, etc. in the requested language. **The game provider should make sure only Anakatech GA IP is whitelisted for this call. GA will validate the request for history with the platform before calling the game provider’s endpoint.**

Function Signature

https://game_provider_url/history

HTTP Method

GET

Input query parameters

Parameter Name	Parameter Type	Required	Description
playerId	String..32 Alphanumeric characters in addition to @ _ - = () # % ! + * ‘	Mandatory	The unique ID of the player's account in the game provider's system. Mandatory parameter except for anonymous play mode.
roundId	an..128	Mandatory	The unique round ID
language	a 2	Mandatory	ISO 639-1 language code. For example: "en", "nl".

Game aggregator – Checksum tool API

In order to get checksums for game provider games, GA will request information either for all checksums for all games, or for specific game. The game provider should implement these 2 endpoints, that will be open only for the GA IP, to return necessary checksums per game.

Get Checksums

Function Signature

https://game_provider_url/getChecksum/{customerId}/{brandId}

HTTP Method

GET

Input path parameters

Parameter Name	Parameter Type	Required	Description
customerId	an..128	Mandatory	The customer ID. Will be provided to you by your account manager.
brandId	an..128	Mandatory	The ID of the customer's brand. Will be provided to you by your account manager.

Output

Parameter name	Parameter Type	Restrictions	Parameter description	Remarks
gameName	String	an..256	The name of the game.	
providerGameId	String	an..256	The unique game ID.	
brandId	String	an..128	The ID of the customer's brand. Will be provided to you by your account manager.	
customerId	String	an..128	The customer ID. Will be provided to you by your account manager.	
list	List<GameChecksumData>		The list of checksums per game	Provided below

GameChecksumData

Parameter name	Parameter Type	Restrictions	Parameter description	Remarks
fileName	String	an..256	The name of the file.	
certifiedHash	String	an..256	The Hash of the file that has been certified.	Calculated using SHA-1 algorithm, hashing inside of a file.
actualHash	String	an..128	The actual Hash of the file.	Calculated using SHA-1 algorithm, hashing inside of a file.
status	String	an..8	“OK” or “MISMATCH”	If certifiedHash and actualHash are equal, then “OK”, else – “MISMATCH”

```
[
  {
    "gameName" : "Kettle1",
    "providerGameId" : "1012",
    "brandId" : "1",
    "customerId" : "1"
    "list" : [
      {
        "fileName" : "GameFile.java"
        "certifiedHash" : "E35AC31B7ECAC280592849A44C8C0C11824DD98D"
        "actualHash" : "E35AC31B7ECAC280592849A44C8C0C11824DD98D"
        "status" : "OK"
      },
      {
        "fileName" : "AnotherGameFile.java"
        "certifiedHash" : "70799F5DBCA4B31F26CC6CB453B9F4A09D3EBC96"
        "actualHash" : "EA0F29FD032D87D5E0B0AD531267E59156440CA5"
        "status" : "MISMATCH"
      },
      ...
    ]
  },
  {
    "gameName" : "Kettle2",
    ...
  }
]
```

Get Checksums By Game ID

Function Signature

Https://game_provider_url/getChecksum/{customerId}/{brandId}/{providerGameId}

HTTP Method

GET

Input path parameters

Parameter Name	Parameter Type	Required	Description
customerId	an..128	Mandatory	The customer ID. Will be provided to you by your account manager.
brandId	an..128	Mandatory	The ID of the customer's brand. Will be provided to you by your account manager.
providerGameId	an..128	Mandatory	The unique game ID.

Output

Parameter name	Parameter Type	Restrictions	Parameter description	Remarks
gameName	String	an..256	The name of the game.	
providerGameId	String	an..256	The unique game ID.	
brandId	String	an..128	The ID of the customer's brand. Will be provided to you by your account manager.	
customerId	String	an..128	The customer ID. Will be provided to you by your account manager.	
list	List<GameChecksumData>		The list of checksums per game	Provided below

GameChecksumData

Parameter name	Parameter Type	Restrictions	Parameter description	Remarks
fileName	String	an..256	The name of the file.	
certifiedHash	String	an..256	The Hash of the file that has been certified.	Calculated using SHA-1 algorithm, hashing inside of a file.
actualHash	String	an..128	The actual Hash of the file.	Calculated using SHA-1 algorithm, hashing inside of a file.
status	String	an..8	"OK" or "MISMATCH"	If certifiedHash and actualHash are equal, then "OK", else – "MISMATCH"

Example:

```
{
  "gameName" : "Kettle1",
  "providerGameId" : "1012",
  "brandId" : "1",
  "customerId" : "1"
  "list" : [
    {
      "fileName" : "GameFile.java"
      "certifiedHash" : "E35AC31B7ECAC280592849A44C8C0C11824DD98D"
      "actualHash" : "E35AC31B7ECAC280592849A44C8C0C11824DD98D "
      "status" : "OK"
    },
    {
      "fileName" : "AnotherGameFile.java"
      "certifiedHash" : "70799F5DBCA4B31F26CC6CB453B9F4A09D3EBC96 "
      "actualHash" : "EA0F29FD032D87D5E0B0AD531267E59156440CA5 "
      "status" : "MISMATCH"
    },
    ...
  ]
}
```

Game aggregator – Free Rounds API

In order to issue free rounds for game provider games, GA will send request for issuing and cancelling free rounds. The game provider should implement these 2 endpoints, that will be open only for the GA IP, to return necessary checksums per game.

Issue FreeRounds

Function Signature

https://game_provider_url/freeRounds/issue

HTTP Method

POST

Input

Parameter name	Parameter Type	Restrictions	Required	Parameter description	Remarks
playerId	String	an..32	Mandatory	The ID of the player's account in the game provider's system	Uniquely identifies a player account per brand.
freeRoundsInstanceId	String	an..32	Mandatory	The unique free round instance id.	

gamesList	String	an..1024	Mandatory	The list of gameIds in a string, separated with “,”.	
gameProviderId	int	n 3	Mandatory	The unique ID of the provider	Will be provided to you by your account manager
betPerLine	int	n 8	Mandatory	The amount of bet per line.	
currency	String	an..3	Mandatory	The ISO 4217 code	e.g. EUR
linesNum	int	an..256	Mandatory	The number of lines.	
numberOfRounds	int	an..256	Mandatory	The number of free rounds per game.	
expirationDate	Date	a..36	Mandatory	The expiration date of free rounds.	
customerId	String	an..8	Mandatory	The customer ID. Will be provided to you by your account manager.	
brandId	String	an..8	Mandatory	The ID of the customer’s brand. Will be provided to you by your account manager.	

Example:

```
{
  "playerId" : "8193449",
  "freeRoundsInstanceId" : "3933",
  "gamesList" :
  "16350,16352,16370,16464,16358,16460,16392,16390",
  "gameProviderId" : 16,
  "betPerLine" : 100,
  "currency" : "EUR",
  "linesNum" : 1,
  "numberOfRounds" : 10,
  "expirationDate" : "Sun Aug 11 07:00:00 UTC 2024",
  "customerId" : "10",
  "brandId": "8"
}
```

Output

Parameter name	Parameter Type	Restrictions	Parameter description	Remarks
success	Boolean		true in case the request succeeded, false otherwise.	See Appendix B for detailed output structure in case of errors.
errorCode	String	an..32	An error code describing the reason for the failure	See Appendix B for detailed output structure in case of errors.
freeRoundsInstanceId	String	an..32	The unique free round instance id.	

Example:

```
{
  "success" : true,
  "freeRoundsInstanceId" : "3933"
}
```

Cancel FreeRounds

Function Signature

Https://game_provider_url/freeRounds/cancel

HTTP Method

POST

Input

Parameter name	Parameter Type	Restrictions	Required	Parameter description	Remarks
customerId	String	an..8	Mandatory	The customer ID. Will be provided to you by your account manager.	
cancelFreeRoundsInstances	Object[]		Mandatory	A list of free round cancel instances.	

cancelFreeRoundsInstance

Parameter name	Parameter Type	Restrictions	Required	Parameter description	Remarks
playerId	String	an..32	Mandatory	The ID of the player's account in the game provider's system	Uniquely identifies a player account per brand.
freeRoundsInstanceId	String	an..32	Mandatory	The unique free round instance id.	
gameProviderId	int	n 3	Mandatory	The unique ID of the provider	Will be provided to you by your account manager
providerGameId	String	an..256	Mandatory	The game ID	Each game has a unique ID.
brandId	String	an..8	Mandatory	The ID of the customer's brand. Will be provided to you by your account manager.	

Example:

```
{
  "customerId" : "10",
  "cancelFreeRoundsInstances" : [{
    "playerId" : "8193449",
    "freeRoundsInstanceId" : "3933",
  }]
}
```

```

        "gameProviderId" : 16,
        "providerGameId" : "16350",
        "brandId": "8"
    }, {
        "playerId" : "8193449",
        "freeRoundsInstanceId" : "3934",
        "gameProviderId" : 16,
        "providerGameId" : "16351",
        "brandId": "8"
    }...
    {}
}

```

Output (list)

Parameter name	Parameter Type	Restrictions	Parameter description	Remarks
success	Boolean		true in case the request succeeded, false otherwise.	See Appendix B for detailed output structure in case of errors.
errorCode	String	an..32	An error code describing the reason for the failure	See Appendix B for detailed output structure in case of errors.
freeRoundsInstanceId	String	an..32	The unique free round instance id.	

Example:

```

[ {
    "success" : true,
    "freeRoundsInstanceId" : "3933"
  }, {
    "success" : true,
    "freeRoundsInstanceId" : "3934"
  }...{} ]

```

Regulations

The game aggregator game providers can support various regulations. Supporting a regulation means the game provider API \ Game clients are compatible with the regulation requirements within this regulation.

Every launch of the game aggregator games requires a parameter with the target jurisdiction.

The game aggregator keeps a mapping (see [GameProviderJurisdictions](#) table) of each game provider and their compatible jurisdictions.

UKGC - RTS 13

According to the gambling commission standard [document](#), A Reality check mechanism is required in this jurisdiction.

To enable reality check on a game - use the realityCheckInterval, realityCheckStartTime parameters while [launching the game](#).

Appendix B - Game Aggregator error handling

All universal API calls may result in an error for various reasons. Responses for erroneous API requests should still return HTTP status “200” (OK) but with the following json in the body:

Parameter name	Parameter Type	Restrictions	Required	Parameter description	Remarks
success	Boolean		Mandatory	Will be set to false in case of failure	
errorCode	string	256	Mandatory	An error code describing the reason for the failure	See the table of possible error codes below.
balance	double	n..32	Mandatory		
cashBalance	double	n..32	Mandatory	The player's account cash balance in cents	Used in regulated markets
bonusBalance	double	n..32	Mandatory	The player's account bonus balance in cents	Used in regulated markets

The table below lists the possible errors which can be returned by the game provider to the Universal API method calls:

Error Code	Description	API Methods	Explanation
GA_501	Internal error	All universal API methods	Returned in cases where a general error occurred (e.g. a software bug)
GA_502	Invalid argument	All universal API methods	For all cases where the provided arguments are invalid (e.g. bet is missing in the request).
GA_503	Invalid secret	All universal API methods	In case the Game aggregator provided an incorrect secret.
GA_504	Untrusted IP	All universal API methods	In case the Game aggregator request's IP is not trusted.
GA_601	Invalid player ID or session ID	All universal API methods	Returned in case the player's ID or session ID is invalid (e.g. session expired\corrupted)
GA_602	Insufficient balance	debit \ debitAndCredit	Not enough balance to cover the bet.
GA_603	Player limits exceeded	debit \ debitAndCredit	Returned in case the player exceeded limits
GA_604	Currency not permitted	credit, debit \ debitAndCredit	Returned in case the provided currency is not permitted for the player.
GA_605	Round does not exist	credit, rollbackDebit	Returned in case the Game aggregator updates a transaction of a non existing roundId. (e.g. a transaction that no prior 'debit' with the roundId was initiated for it)
GA_606	Round was already closed	credit, debit	Returned if a debit \ credit request for a new transaction is associated to an already closed round id (closeRound = 1 request was received earlier for this roundId)

Appendix C - API additionalData JSON

In additionalData there will be additional information regarding freeRounds:

Input

Parameter name	Parameter Type	Required	Parameter description	Remarks
freeRoundsPlay	Boolean	Optional	Provided only when free rounds exists	If active free rounds exist it will be set to true
freeRoundsBet	int	Optional	Provided only when free rounds exists	If active free rounds exist it will hold the free rounds bet <u>per line (in cents)</u> , but in case of no lines, it will be the total bet.
totalFreeRounds	int	Optional	Provided only when free rounds exists	If active free rounds exist it will hold the number of free rounds
playedFreeRounds	int	Optional	Provided only when free rounds exists	If active free rounds exist it will hold the number of already played free rounds
freeRoundsWinnings	int	Optional	Provided only when free rounds exists	If active free rounds exist it will hold the total winnings amount (in cents) from free rounds
freeRoundsId	String	Optional	Provided only when free rounds exists	If active free rounds exist it will hold the free round ID.