Spadegaming API

Specifications Document

(English Version)

Version 1.4

Revision History

V1.0

- First release of the new version document
  - Combine Balance Transfer and Common Wallet API documents
  - Remove authorize function, add getAuthorize
  - Added new API: getTicketLog

V1.1

- Add new Param on getAuthorize Launch E-Game by API : balance
- Add new Param on KickAcct: gameCode
- Remove Param on KickAcct : list[]

V1.2

- Add spinAll example at Appendix 6
- refTicketIds enchancement
- enchance categoryId at getBetHistory.

V1.3

- Add new Param on getTicketLog : allTransferTicketLog
- Remove Param on getTicketLog : currTransferTicketLog

V1.4

- Add new API: getMemberList
- Add new language : Portuguese (pt_PT)

Contents

1. Protocol

HTTP(s) are used to send and receive data. Data of type XML and JSON are supported. Refer to the http://www.json.org/ for more information of JSON.

In additional to the standard HTTP protocol, the following HTTP Headers are required:

HTTP Header	Mandatory	Example	Description
API	Yes	authorize	Service Name
DataType	Yes	JSON	Data Type (XML or JSON)
Digest	Yes	MD5 (Data+securityKey)	Message digest of the post data -securityKey will provide by Game Provider (SG)
Accept-Encoding	No	gzip, deflate	Identify whether or not support gzip compression of the returning data
Content-Encoding	No	gzip	Identify whether or not using gzip compression of the post data
Accept-Language	No	en_US	Set the default language of the returning data.

Note:

Please use UTF-8 as the default encoding for all encode/decode codes when convert text to/from byte[], without any spaces and line breaks .

```
Public abstract class DigestUtils
{
public static String digest(byte[] input) {
byte[] digest = Digest.md5().compute( input );
return Hex.encodeHexString( digest );
  }

public static String digest(byte[] input, byte[]secretKey) {
int bodySize=input.length,keySize=secretKey.length;
byte[] buffer= new byte[bodySize+keySize];
System.arraycopy(input,0,buffer,0,bodySize);
System.arraycopy(secretKey,0,buffer,bodySize,keySize);
byte[] digest = Digest.md5().compute( input );
return Hex.encodeHexString(digest);
  }
}
```

2. Glossary

Game Provider	Spadegaming game provider
Merchant	The party that wants to integrate Spadegaminggames
API Requestor	The party responsible for requesting the function
API Provider	The party responsible for responding the function
Acct ID	Player's unique identifier
Request	JSON/XML encoded request
Response	JSON/XML encoded response

3. API Interface Overview

During API call, the invoker will create the request message according to API specification, and post the data to the URL. API provider will send the response message after processing the request.

Usually the message will be defined as：xxxResponse xxxx(xxxRequest request)

Where *xxxRequest* equates to API request message and *xxxResponse* equates to API response message.

All request messages will contain the following fields:

Name	**Data Type**	**Example**	**Description**
serialNo	Varchar(50)	20120722224255982841	generated by "requestor"
merchantCode	Varchar(10)	SPADE	Unique code that identifies the Merchant.

All response messages will contains the following fields:

Name	**Data Type**	**Example**	**Description**
serialNo	Varchar(50)	20120722224255982841	Same as request's serialNo
code	Integer	0	Processed result status.
msg	Varchar(128)	success	Processed result description.
merchantCode	Varchar(10)	SPADE	Unique code that identifies the Merchant.

Please refer to Appendix 1 for the list of status codes that system returns.

Warning: serialNo is the identity of the message, and one serialNo for one request or response. Please make sure the serialNo is unique.

We advise use GUID (UUID), or the date+time+random string as the serialNo

4. API Interface Definition

4.1 Launch Game Lobby by API

4.1.1 Launch E-Game by API

Usage: Player enters E-games page by API

Interface Name： getAuthorize
API Provider： Game Provider
API Requestor： Merchant

Request Message Definition for `getAuthorizeRequest`:

Data Type	Mandatory	Example	Description
acctInfo	Yes	AcctInfoType	acctInfo

AcctInfo type definition：

Name	Data Type	Mandatory	Example	Description
acctId	Varchar(50)	Yes	TESTPLAYER1	Unique player ID
userName	Varchar(15)	No	TESTPlayer	Player name
siteId	Varchar(10)	No	SITE_USD1	Unique site ID for merchant
currency	Char(3)	Yes	USD	Currency Code
balance	Decimal(20,4)	No	100	Current youtuber player balance **Term Condition Used:** Common Wallet + Youtuber Type

Name	Data Type	Mandatory	Example	Description
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code
token	Varchar(80)	Yes	fe1a85adc54545d2963b661a22d09c9e	
acctIp	Varchar(50)	Yes	8.8.8.8	AcctId IP
game	Varchar(10)	Yes	S-LK03	game code
language	Varchar(10)	No	en_US	Language code.
exitUrl	Varchar(150)	No	http://www.domain.example.com	Redirect Url to exit button
serialNo	Varchar(50)	No	20120722224255982841	Generated by merchant to indicate message sequence.
mobile	Boolean(10)	No	true/false	Play in mobile
fun	Boolean (10)	No	true / false	play for fun or real money
menuMode	Boolean (10)	No	true/false	Game Menu Switch off
fullScreen	Boolean(10)	No	true/false	Full screen Switch off

Response message definition for `getAuthorizeResponse`:

Name	Data Type	Mandatory	Example	Description
token	Varchar(80)	Yes	a3b0c9dd1ab041	Unique token for

				merchant
serialNo	Varchar(50)	Yes	498036704	Merchant transaction id
gameUrl	Varchar(200)	Yes		Game url
msg	Varchar(50)	Yes	success	
code	Integer	Yes	0	Response Code

Note:

1. For language, Refer to Appendix 2.
2. If play for fun, the URL must contains acctId and token, fun=true.
3. If play for mobile game, must contains mobile=true
4. If not required game menu, must contains menuMode=false
5. If not required full screen, must contains fullScreen=false
6. Refer to Appendix 3 for a list of currency code that system support.
7. userName is used to display on the lobby
8. Acct ID format: [A-Z, 0-9,_-@]{1,50}. Example: TESTPLAYER1

getAuthorize (Request and response messages example):

JSON getAuthorizeRequest：

```
{
      "merchantCode": "SPADE",
      "acctInfo": {
            "acctId": "TESTPLAYER1",
            "userName": "TESTPlayer",
            "currency": "USD",
            "siteId": "SITE_USD1",
            "balance":10000
      },
      "language": "en_US",
      "token": "fe1a85adc54545d2963b661a22d09c9e",
      "game": "S-LK03",
      "acctIp": "8.8.8.8",
      "fun": "false",
      "mobile": "true",
      "menuMode": "true",
      "exitUrl": "http://www.domain.example.com ",
      "fullScreen": "true",
      "serialNo": "20120722224255982841"
}
```

XML getAuthorizeRequest：

```
<getAuthorizeRequest>
   <acctInfo>
      <acctId>TESTPLAYER1</acctId>
      <userName>TESTPLAYER1</userName>
```

```
      <currency>USD</currency>
      <siteId>SITE_USD1</siteId>
     <balance>10000</balance>
   </acctInfo>
   <token>fe1a85adc54545d2963b661a22d09c9e</token>
   <merchantCode>SPADE</merchantCode>
   <game>S-LK03</game>
   <fun>false</fun>
   <mobile>true</mobile>
   <menuMode>true</menuMode>
   <exitUrl>http://www.domain.example.com</exitUrl>
   <fullScreen>true</fullScreen>
   <acctIp>8.8.8.8</acctIp>
   <serialNo>20120722224255982841</serialNo>
</getAuthorizeRequest>
```

JSON getAuthorizeResponse：

```
{
      "token": "fe1a85adc54545d2963b661a22d09c9e",
      "gameUrl": "
https://example.domain.com/auth/?acctId=TESTPLAYER1&language=en_US
&token=fe1a85adc54545d2963b661a22d09c9e&game=S-
LK03&fun=false&mobile=true&menuMode=on&exitUrl=http://www.domain.e
xample.com",
      "msg": "success",
      "code": 0,
      "serialNo": "20120722224255982841"
}
```

XML getAuthorizeResponse:

```
<getAuthorizeResponse>
   <serialNo>20120722224255982841</serialNo>
   <code>0</code>
   <msg>success</msg>
   <token>fe1a85adc54545d2963b661a22d09c9e</token>
   <gameUrl>
https://example.domain.com/auth/?acctId=TESTPLAYER1&language=en_US
&token=fe1a85adc54545d2963b661a22d09c9e&game=S-
LK03&fun=false&mobile=true&menuMode=on&exitUrl=http://www.domain.e
xample.com</gameUrl>
</getAuthorizeResponse>
```

4.1.2 Launch Game Lobby by API

Usage: Player enters SG lobby by API

Interface Name：getAuthorize
API Provider：Game Provider
API Requestor：Merchant

Request Message Definition for `getAuthorizeRequest`:

Data Type	Mandatory	Example	Description
acctInfo	Yes	AcctInfoType	acctInfo

AcctInfo type definition:

Name	Data Type	Mandatory	Example	Description
acctId	Varchar(50)	Yes	TESTPLAYER1	Unique player ID
userName	Varchar(15)	No	TESTPlayer	Player name
siteId	Varchar(10)	No	SITE_USD1	Unique site ID for merchant
currency	Char(3)	Yes	USD	Currency Code

Name	Data Type	Mandatory	Example	Description
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code
token	Varchar(80)	Yes	fe1a85adc54545d2963b661a22d09c9e	
acctIp	Varchar(50)	Yes	8.8.8.8	AcctId IP
Lobby	Varchar(10)	Yes	SG	Lobby Code
language	Varchar(10)	No	en_US	Language code.
serialNo	Varchar(50)	No	20120722224255982841	Generated by merchant to indicate message sequence.
mobile	Boolean(10)	No	true/false	Play in mobile

Response message definition for `getAuthorizeResponse`:

Name	Data Type	Mandatory	Example	Description
token	Varchar(80)	Yes	a3b0c9dd1ab041	Unique token for merchant
serialNo	Varchar(50)	Yes	498036704	Merchant transaction id
gameUrl	Varchar(200)	Yes		Game url
msg	Varchar(50)	Yes	success	
code	Integer	Yes	0	Response Code

Note:

1. For language, Refer to Appendix 2.
2. If play for mobile game, must contains mobile=true
3. Refer to Appendix 3 for a list of currency code that system support.
4. userName is used to display on the lobby
5. Acct ID format: [A-Z, 0-9,_-@]{1,50}. Example: TESTPLAYER1
6. Lobby no support Fun for Play

getAuthorize (Request and response messages example):

JSON getAuthorizeRequest：

```
{
   "merchantCode":"SPADE",
   "acctInfo":{
      "acctId":"TESTPLAYER1",
      "userName":"TESTPlayer",
      "currency":"USD",
      "siteId":"SITE_USD1"
   },
   "language":"en_US",
   "token":"fe1a85adc54545d2963b661a22d09c9e",
   "lobby":"SG",
   "acctIp":"8.8.8.8",
   "mobile":"true",
   "serialNo":"20120722224255982841"
}
```

XML getAuthorizeRequest：

```
<getAuthorizeRequest>
   <acctInfo>
      <acctId>TESTPLAYER1</acctId>
      <userName> TESTPLAYER1</userName>
      <currency>USD</currency>
      <siteId>SITE_USD1</siteId>
   </acctInfo>
   <token>fe1a85adc54545d2963b661a22d09c9e</token>
   <merchantCode>SPADE</merchantCode>
   <game>S-LK03</game>
   <fun>false</fun>
   <mobile>true</mobile>
   <acctIp>8.8.8.8</acctIp>
   <serialNo>20120722224255982841</serialNo>
</getAuthorizeRequest>
```

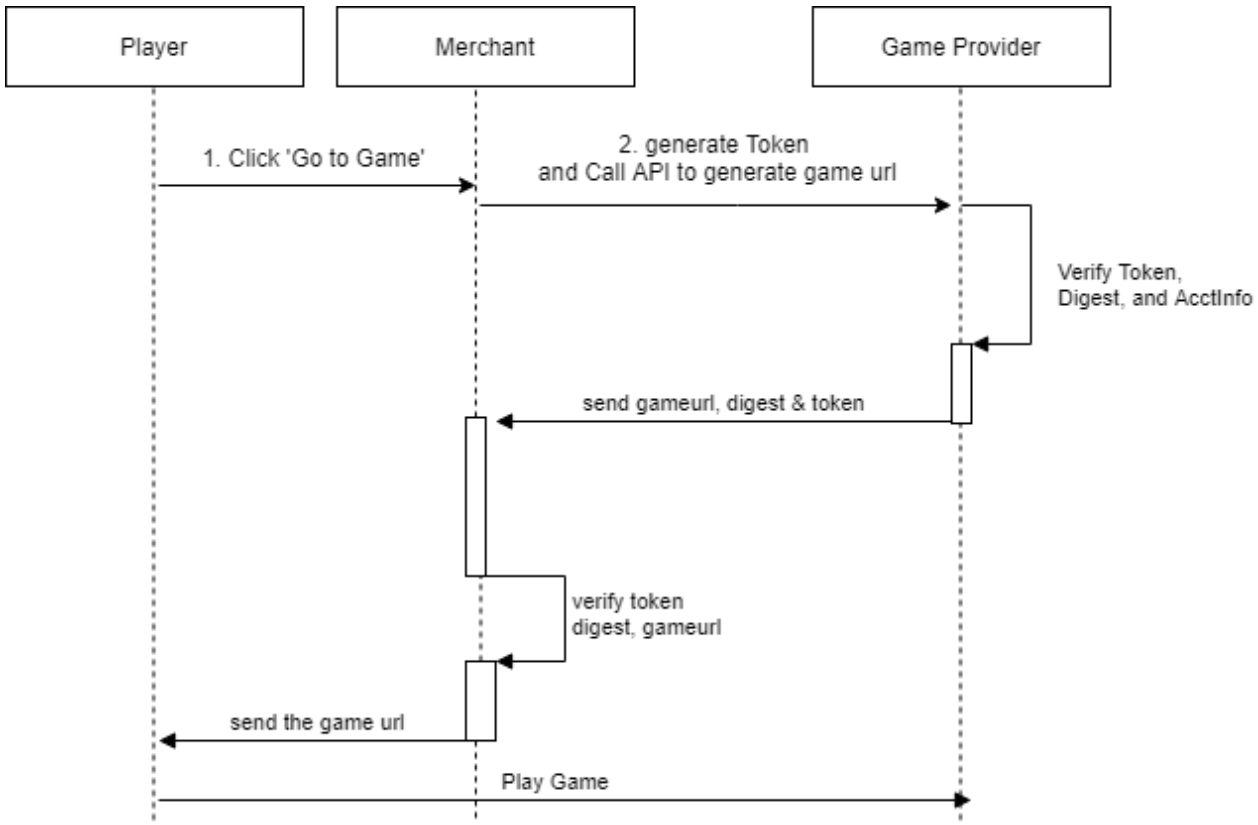
JSON getAuthorizeResponse：

```
{
   "token":"fe1a85adc54545d2963b661a22d09c9e",
   "gameUrl":"
https://example.domain.com/lobby/SG/?acctId=TESTPLAYER1&language=e
n_US&token=fe1a85adc54545d2963b661a22d09c9e&lobby=SG",
   "msg":"success",
   "code":0,
   "serialNo":"20120722224255982841"
}
```

XML getAuthorizeResponse:

```
<getAuthorizeResponse>
   <serialNo>20120722224255982841</serialNo>
   <code>0</code>
   <msg>success</msg>
   <token>fe1a85adc54545d2963b661a22d09c9e</token>
   <gameUrl>
https://example.domain.com/lobby/SG/?acctId=TESTPLAYER1&language=e
n_US&token=fe1a85adc54545d2963b661a22d09c9e&lobby=SG</gameUrl>
</getAuthorizeResponse>
```

4.1.3 Launch Game Flow Chart



4.2 Balance Transfer

4.2.1 Query user information

Usage scenario: Search for user information (Balance & etc.)

Interface Name：getAcctInfo

API Provider：Game Provider

API Requestor：Merchant

Request Message Definition for `GetAcctInfoRequest`:

Name	Data Type	Mandatory	Example	Name
acctId	Varchar(50)	No	TESTPLAYER1	Unique player ID, leave 'Empty' to return all players.
pageIndex	Integer	Yes	1	Page index
serialNo	Varchar(50)	Yes	20120722224255982841	generated by "requestor"
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code

Response message definition for `GetAcctInfoResponse`:

Name	Data Type	Mandatory	Example	Name
resultCount	Integer	Yes	100	Total record count
pageCount	Integer	Yes	1	Total page count
list	AcctInfo[]	Yes		AcctInfo Type

AcctInfo type definition

Name	Data Type	Mandatory	Example	Name
acctId	Varchar(50)	Yes	TESTPLAYER1	Unique player ID
userName	Varchar(15)	No	TESTPlayer1	Player name
currency	Char(3)	Yes	USD	Currency Code
balance	Decimal(20,4)	Yes	1000	Current player balance

GetAcctInfo (Request and response messages example):

JSON GetAcctInfoRequest：

```
{
    "acctId": "TESTPLAYER1",
    "pageIndex": 0,
    "merchantCode": "SPADE",
    "serialNo": "20120722230043734928"
}
```

XML GetAcctInfoRequest:

```
<GetAcctInfoRequest>
      <serialNo>20120802152140143938</serialNo>
      <merchantCode>SPADE</merchantCode>
      <pageIndex>0</pageIndex>
      <acctId>TESTPLAYER1</acctId>
</GetAcctInfoRequest>
```

JSON GetAcctInfoResponse：

```
{
      "list": [{
            "userName": "TESTPlayer1",
            "currency": "USD",
            "acctId": "TESTPLAYER1",
            "balance": 1000
      }],
      "resultCount": 1,
      "pageCount": 1,
      "merchantCode": "SPADE",
      "msg": "success",
      "code": 0,
      "serialNo": "20120802152140143938"
}
```

XML GetAcctInfoResponse:

```
<GetAcctInfoResponse>
      <serialNo>20120802152140143938</serialNo>
      <code>0</code>
      <msg>0</msg>
      <merchantCode>SPADE</merchantCode>
      <resultCount>0</resultCount>
      <pageCount>1</pageCount>
      <list>
            <AcctInfo>
                  <acctId>TESTPLAYER1</acctId>
                  <userName>TESTPlayer1</userName>
                  <currency>USD</currency>
                  <balance>1000.0000</balance>
            </AcctInfo>
      </list>
</GetAcctInfoResponse>
```

4.2.2 Deposit

Usage：When merchant wants to transfer money to Game provider wallet for a particular member.

Interface Name：deposit

API Provider：Game Provider
API Requestor：Merchant

Request Message Definition for `DepositRequest`:

Name	Data Type	Mandatory	Example	Description
acctId	Varchar(50)	Yes	TESTPLAYER1	Unique player ID
currency	Char(3)	Yes	USD	Currency Code
amount	Decimal(20,4)	Yes	1000	Amount to be added to player balance.
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code
serialNo	Varchar(50)	Yes	20120722231413595602	generated by "requestor"

Response message definition for `DepositResponse`:

Name	Data Type	Mandatory	Example	Description
transactionId	Varchar(20)	Yes	498036704	System generated unique id.
afterBalance	Decimal(20,4)	Yes	1000.0000	Player's latest balance

Notes:

1. SerialNo is the identity of the message, and one serialNo for one request or response. Please make sure the serialNo is unique for deposit and withdraw message
2. We advise to use GUID (UUID), or the date+time+random string as the serialNo
3. Please resend the previous original request, if do not received the transfer request from our API server,
   - Must make sure the serialNo of the request message is same as previous.
   - We will process the transfer (deposit & withdraw) only 1 time for 1 serialNo.
4. Acct ID format: [A-Z,0-9,-@]{1,50}. Example: TestPlayer_1

Deposit (Request and response messages example):

JSON DepositRequest：

```
{
   "acctId":"TESTPLAYER1",
   "amount":200,
   "currency":"USD",
   "merchantCode":"SPADE",
   "serialNo":"20120722231413699735"
}
```

XML DepositRequest：

```
<DepositRequest>
      <serialNo>20120722231413699735</serialNo>
      <merchantCode>SPADE</merchantCode>
      <acctId>TESTPLAYER1</acctId>
      <currency>USD</currency>
      <amount>200</amount>
</DepositRequest>
```

JSON DepositResponse：

```
{
   "transactionId":"500648408",
   "merchantCode":"SPADE",
   "afterBalance":1000.0000,
   "msg":"success",
   "code":0,
   "serialNo":"20120722231413699735"
}
```

XML DepositResponse：

```
<DepositResponse>
   <serialNo>20120722231413699735</serialNo>
   <code>0</code>
   <msg>success</msg>
   <merchantCode>SPADE</merchantCode>
   <transactionId>500648408</transactionId>
   <afterBalance>1000.0000</afterBalance>
</DepositResponse>
```

4.2.3 Withdrawal

Usage：When merchant wants to transfer money from Game provider wallet for a particular member.

Interface Name：withdraw

API Provider：Game Provider
API Requestor：Merchant

Request Message Definition for `WithdrawRequest`:

Name	Data Type	Mandatory	Example	Description
acctId	Varchar(50)	Yes	TESTPLAYER1	Unique player ID
currency	Char(3)	Yes	USD	Currency Code
amount	Decimal(20,4)	Yes	1000	Amount to be reduced from player balance.
serialNo	Varchar(50)	Yes	20120722231413595602	generated by "requestor"
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code

Response message definition for `WithdrawResponse`:

Name	Data Type	Mandatory	Example	Description
transactionId	Varchar	Yes	498036704	System generated unique id.
afterBalance	Decimal(20,4)	Yes	1000.0000	Player's latest balance

Note:

1. SerialNo is the identity of the message, and one serialNo for one request or response. Please make sure the serialNo is unique for deposit and withdraw message
2. We advise to use GUID (UUID), or the date+time+random string as the serialNo
3. Please resend the previous original request, if do not received the transfer request from our API server,
   - Must make sure the serialNo of the request message is same as previous.
   - We will process the transfer (deposit & withdraw) only 1 time for 1 serialNo.
4. Acct ID format: [a-zA-Z0-9_-@]{1,50}. Example: TestPlayer_1

Withdraw (Request and response messages example):

JSON WithdrawRequest：

```
{
   "acctId":"TESTPLAYER1",
   "amount":200,
   "currency":"USD",
   "merchantCode":"SPADE",
   "serialNo":"20120722231413595602"
}
```

XML WithdrawRequest：

```
<WithdrawRequest>
   <serialNo>20120722231413595602</serialNo>
   <merchantCode>SPADE</merchantCode>
   <acctId>TESTPLAYER1</acctId>
   <currency>USD</currency>
   <amount>200</amount>
</WithdrawRequest>
```

JSON WithdrawResponse：

```
{
   "transactionId":"498036704",
   "afterBalance":1000.0000,
   "merchantCode":"SPADE",
   "msg":"success",
   "code":0,
   "serialNo":"20120722231413595602"
}
```

XML WithdrawResponse：

```
<WithdrawResponse>
   <serialNo>20120722231413595602</serialNo>
   <code>0</code>
   <msg>success</msg>
   <merchantCode>SPADE</merchantCode>
   <transactionId>498036704</transactionId>
   <afterBalance>1000.0000</afterBalance>
</WithdrawResponse>
```

4.2.4 Query deposit/withdraw API call status

Usage：Merchant wishes to check the status of a deposit/withdraw particular API Call

Interface Name：checkStatus

API Provider：Game Provider
API Requestor：Merchant

Request Message Definition for `CheckStatusRequest`：

Name	Data Type	Mandatory	Example	Description
lastSerialNo	Varchar(50)	Yes	20120723175139440637	Withdraw / Deposit serialNo
serialNo	Varchar(50)	Yes	20120722231413595602	generated by "requestor"
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code

Response message definition for `CheckStatusResponse`：

Name	Data Type	Mandatory	Example	Description
status	Integer	Yes	0	Status code 0=success ; 1=failed 110= Not found
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code
serialNo	Varchar(50)	Yes	20120722231413595602	Same as request's serialNo
code	Integer	Yes	0	Status code 0=success ; 1=failed

Note:

1. Status is the serialNo status of request
2. lastSerialNo can be retrieve from the previous request of deposit/withdraw serialNo
3. Code is the request Success (0) or Failed (1)

CheckStatus (Request and response messages example):

JSON CheckStatusRequest：

```
{
      "lastSerialNo": "20120726214956319959",
      "merchantCode": "SPADE",
      "serialNo": "20120726214956563472"
}
```

XML CheckStatusRequest：

```
<CheckStatusRequest>
  <serialNo>20120726214956563472</serialNo>
  <merchantCode>SPADE</merchantCode>
  <lastSerialNo>20120726214956319959</lastSerialNo>
</CheckStatusRequest>
```

JSON CheckStatusResponse：

```
{
      "status": 0,
      "merchantCode": "SPADE",
      "msg": "success",
      "code": 0,
      "serialNo": "20120726214956563472"
}
```

XML CheckStatusResponse：

```
<CheckStatusResponse>
   <serialNo>20120726214956563472</serialNo>
   <code>0</code>
   <msg>success</msg>
   <merchantCode>SPADE</merchantCode>
   <status>0</status>
</CheckStatusResponse>
```

4.2.5 Query the Member not on line but have amount

Usage：The member not online but have amount

Interface Name：pendingAcct

API Provider：Game Provider

API Requestor：Merchant

Request message definition for `PendingAcctRequest`:

Name	Data Type	Mandatory	Example	Description
serialNo	Varchar(50)	Yes	20120722231413595602	generated by “requestor"
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code

Response message definition for `PendingAcctResponse`:

Name	Data Type	Mandatory	Example	Description
list	AcctInfo []	yes		AcctInfo

AccountInfo type definition：

Name	Data Type	Mandatory	Example	Description
acctId	Varchar(50)	Yes	TESTPLAYER1	Unique player ID
currency	Char(3)	Yes	USD	Currency Code
balance	Decimal(20,4)	Yes	1000	Current player balance

PendingAcct(Request and response messages example):

JSON PendingAcctRequest：

```
{
      "merchantCode": "SPADE",
      "serialNo": "20130502191551906534"
}
```

XML PendingAcctRequest:

```
<PendingAcctRequest>
  <serialNo>20130502191551906534</serialNo>
  <merchantCode>SPADE</merchantCode>
</PendingAcctRequest>
```

JSON PendingAcctResponse：

```
{
      "list": [{
            "currency": "USD",
            "acctId": "TESTPLAYER1",
            "balance": 1000
      }],
      "merchantCode": "SPADE",
      "code": 0,
      "msg": "success",
      "serialNo": "20130502191551906534"
}
```

XML PendingAcctResponse:

```
<PendingAcctResponse>
  <serialNo>20130502191551906534</serialNo>
  <code>0</code>
  <msg>success</msg>
  <merchantCode>SPADE</merchantCode>
  <list>
    <PendingAcct>
      <acctId>TESTPLAYER1</acctId>
      <currency>USD</currency>
      <balance>1000</balance>
    </PendingAcct>
  </list>
</PendingAcctResponse>
```

4.2.6 Query Fund In Out

Usage：Search for deposit and withdraw history

Interface Name：fundInOut

API Provider：Game Provider

API Requestor：Merchant

Request Message Definition `FundInOutRequest`:

Name	Data Type	Mandatory	Example	Description
beginDate	Char(15)	Yes	20190401T000000	Start date range to search. Format must be in yyyymmdd'T'24hhmmss
endDate	Char(15)	Yes	20190403T200000	End date range to search. Format must be in yyyymmdd'T'24hhmmss
acctId	Varchar(50)	No	TESTPLAYER1	Player id
currency	Char(3)	No	USD	Currency code
lastSerialNo	Varchar(50)	No	20120722231413595602	Withdraw / Deposit Serial No
type	Varchar	No	In	In = deposit Out = withdraw
pageIndex	Integer	Yes	1	Page number
serialNo	Varchar(50)	Yes	20120722231413595602	generated by "requestor"
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code

Response message definition `FundInOutResponse`:

Name	Data Type	Mandatory	Example	Description
resultCount	Integer	Yes	100	Total record count
pageCount	Integer	Yes	1	Total page count
List	FunInOutInfo[]	Yes		FundInOutInfo Array

FundInOutInfo Definition `FundInOutInfo`:

Name	Data Type	Mandatory	Example	Description
transferId	Varchar(20)	Yes	641482277	Transfer number.
acctId	Varchar(50)	Yes	TESTPLAYER1	Player ID
transactionTime	Char(15)	Yes	20180722T225417	Date time when withdraw or deposit
type	Varchar	Yes	In	In = deposit Out = withdraw
amount	Decimal(20,4)	Yes	1000	Player withdraw/ deposit amount
lastSerialNo	Varchar(50)	Yes	20180722224255982841	Withdraw / Deposit serialNo

FundInOut Request and response messages example:

JSON FundInOut Request:

```
{
      "beginDate":"20190403T000000",
      "endDate":"20190403T200000",
      "acctId":"TESTPLAYER1",
      "currency":"USD",
      "lastSerialNo":”20180722224255982841”
      "type":"In"
      "serialNo":"20180802152140105891"
      "pageIndex":1,
      "merchantCode":"SPADE",
}
```

XML FundInOut Request:

```
<FundInOutRequest>
      <beginDate>20190403T000000</beginDate>
      <endDate>20190403T200000</endDate>
      <acctId>TESTPLAYER1</acctId>
      <currency>USD</currency>
      <lastSerialNo>20180722224255982841</lastSerialNo>
      <type>in</type>
      <serialNo>20180802152140105891</serialNo>
      <pageIndex>1</pageIndex>
      <merchantCode>SPADE</merchantCode>
    </FundInOutRequest>
```

JSON FundInOut Response：


```
{
      "list": [
            {
                  "transferId": 64148227,
                  "acctId": "TESTPLAYER1",
                  "transactionTime": "20180722t225417",
                  "type": "In",
                  "amount": 1000,
                  "lastSerialNo": "201807222224255982841"
            }
      ],
      "resultCount": 1,
      "pageCount": 1,
      "merchantCode": "SPADE",
      "code": 0,
      "msg": "Success",
      "serialNo": "20180802152140105891"
}
```


XML FundInOut Response：


```
<FundInOutResponse>
  <serialNo>20180802152140105891</serialNo>
  <code>0</code>
  <msg>Success</msg>
  <merchantCode>SPADE</merchantCode>
  <resultCount>1</resultCount>
  <pageCount>1</pageCount>
  <list>
    <FundInOutInfo>
      <transferId>64148227</transferId>
      <acctId>TESTPLAYER1</acctId>
      <transactionTime>20180722t225417</transactionTime>
      <type>In</type>
      <amount>1000</amount>
     <lastSerialNo>201807222224255982841</lastSerialNo>
    </FundInOutInfo>
  </list>
</FundInOutResponse>
```

4.3 One Wallet

4.3.1 Query User Balance

Usage scenario: Search for user balance.

Interface Name：getBalance

API Provider：Merchant

API Requestor：Game Provider

Request Message Definition for `GetBalanceRequest`：

Name	Data Type	Mandatory	Example	Remark
acctId	Varchar(50)	Yes	TESTPLAYER1	Acct Id
gameCode	Varchar(10)	No	S-LK03	Game Code
serialNo	Varchar(50)	Yes	20120722224255982841	generated by "Provider"
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code

Response message definition for `GetBalanceResponse`：

Name	Data Type	Mandatory	Example	Remark
acctInfo	AcctInfo	Yes	AcctInfo	acctInfo

AcctInfo type definition：

Name	Data Type	Mandatory	Example	Name
acctId	Varchar(50)	Yes	TESTPLAYER1	Unique player ID
userName	Varchar(15)	No	TESTPlayer1	Player name
currency	Char(3)	Yes	USD	Currency Code
balance	Decimal(20,4)	Yes	1000	Current player balance
siteId	Varchar (10)	No	SITE_USD1	Unique site ID for merchant

GetBalance Request and response messages example:

JSON GetBalanceRequest：

```
{
      "acctId": "TESTPLAYER1",
      "merchantCode": "SPADE",
      "serialNo": "20120722224255982841",
      "gameCode": "S-LK03"
}
```

XML GetBalanceRequest:

```
<GetBalanceRequest>
   <serialNo>20120722224255982841</serialNo>
   <merchantCode>SPADE</merchantCode>
   <acctId>TESTPLAYER1</acctId>
   <gameCode>S-LK03</gameCode>
</GetBalanceRequest>
```

JSON GetBalanceResponse：

```
{
      "acctInfo": {
            "userName": "TestPlayer1",
            "currency": "USD",
            "acctId": "TESTPLAYER1",
            "balance": 1000
      },
      "merchantCode": "SPADE",
      "msg": "success",
      "code": 0,
      "serialNo": "20120722224255982841"
}
```

XML GetBalanceResponse:

```
<GetBalanceResponse>
   <serialNo>20120722224255982841</serialNo>
   <code>0</code>
   <msg>0</msg>
   <merchantCode>SPADE</merchantCode>
   <acctInfo>
      <acctId>TESTPLAYER1</acctId>
      <userName>TestPlayer1</userName>
      <currency>USD</currency>
      <balance>1000.0000</balance>
   </acctInfo>
</GetBalanceResponse>
```

4.3.2 Transfer

Usage：

1) When player place bet，Game Provider will send the transfer request to merchant，and merchant will deduct the player balance.
2) When game provider payout，Game Provider will send the transfer request to merchant，and merchant will increase the player balance.

Interface Name：transfer

API Provider：Merchant

API Requestor：Game Provider

Request Message Definition for `TransferRequest`：

Name	Data Type	Mandatory	Example	Description
transferId	Varchar(50)	Yes	a3b0c9dd1ab041	Unique transfer Id
acctId	Varchar(50)	Yes	TESTPLAYER1	Player id
currency	Char(3)	Yes	USD	Currency code
amount	Decimal(20,9)	Yes	1000	Transfer amount (must greater than or equal to 0)
type	Int	Yes	1	1 = place bet 2 = cancel bet 4 = payout
channel	Varchar(10)	Yes	Web/Mobile, APP-i/APP-A/PC	Ticket for mobile or web
gameCode	Varchar(10)	Yes	S-LK03	Game Code
ticketId	Varchar(20)	No	641482277	
referenceId	Varchar(50)	No	a8ab9cc8f1a277de	Reference to place bet transfer id
specialGame	SpecialGame		SpecialGame Type	Only get bonus, free, bonusfree, freebonus will send this
refTicketIds	Varchar(2048)	No	[120001, 120005]	Reference ticket ids send in interval format Example: [120001, 120005] Notes: Above example included 5 tickets starting from 120001, 120002, 120003, 120004 and 120005
playerIp	Varchar(50)	No	8.8.8.8	Player IP
gameFeature	Varchar(50)	No	BUY-8	Only available when player trigger "Buy Feature"
transferTime	Char(15)	No	20120720T230043	Format yyymmdd’T’24hhmmss

SpecialGame type definition:

Name	Data Type	Mandatory	Example	Description
type	Varchar(20)	No	Free	Differentiate the special game type. Example: bonus, free, bonusfree， freebonus
count	Int	No	10	Total count on the special game
sequence	Int	No	1	Count on the completed special game

specialGame & refTicketIds only will pass during type = 4, payout

Important Note:

Type 4: Payout

1. If the transfer Id received is duplicated, merchant need to check the transfer status and reply the status to game provider. Merchant **CANNOT** process the transfer with same transfer id, just respond us success and take no action to player balance

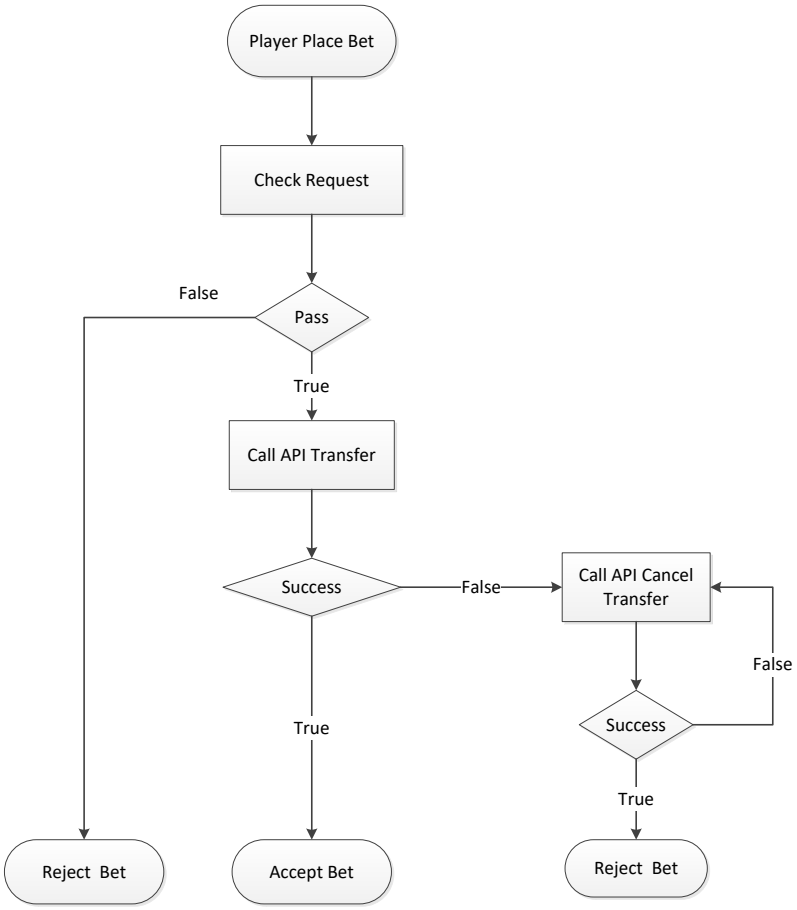
Type 2: Cancel Bet

1. If the reference Id are duplicated or no found on the cancel bet (type=2), Merchant should return us Error Code: 109. Please response us code 0 or 109. Except this error code, provider will continue resend you the reference id.
2. If placeBet encounter timeout (our system did not receive any response), cancel bet will be sent.
3. Even with successful placeBet, under certain circumstances also able to trigger cancel bet. This will happen when the player enters the fishing game and leave the game without any betting (shooting).

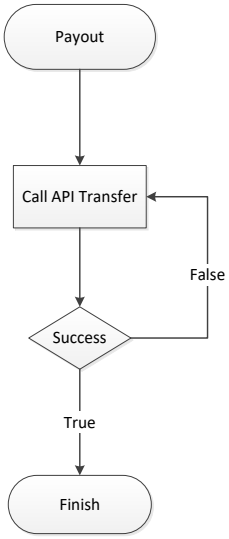
Response message definition for `TransferResponse`:

Name	Data Type	Mandatory	Example	Description
transferId	Varchar(50)	Yes	a3b0c9dd1ab041	Unique transfer Id
merchantTxId	Varchar(20)	No	498036704	Merchant transaction id
acctId	Varchar(50)	Yes	TESTPLAYER1	Player ID
balance	Decimal(20,4)	Yes	1000	Player balance after transfer

Place Bet Flow:



Payout Flow:



Transfer Request and response messages example:

JSON TransferRequest：

```
{
      "acctId": "TESTPLAYER1",
      "transferId": "0ab9bdca06c14811b24653468e60988",
      "currency": "USD",
      "amount": 10,
      "type": 4,
      "ticketId": "234950357",
      "channel": "Web",
      "gameCode": "S-LK03",
      "merchantCode": "SPADE",
      "serialNo": "20120722231413699735",
      "referenceId": "dfffdca06c14811b24653468e60",
      "playerId": "8.8.8.8",
      "gameFeature": "BUY-8",
      "transferTime": "20120720T230043",
      "specialGame": {
            "type": "Free",
            "count": 10,
            "sequence": 1
      },
      "refTicketIds": [
            "120001, 120002, 120003"
      ]
}
```

XML TransferRequest：

```
<TransferRequest>
   <serialNo>20120722231413699735</serialNo>
   <merchantCode>SPADE</merchantCode>
   <transferId>0ab9bdca06c14811b24653468e60988</transferId>
   <acctId>TESTPLAYER1</acctId>
   <currency>USD</currency>
   <amount>10</amount>
   <type>4</type>
   <ticketId>234950357</ticketId>
   <channel>Web</channel>
   <gameCode>S-LK03</gameCode>
   <referenceId>dfffdca06c14811b24653468e60</referenceId>
   <playerId>8.8.8.8</playerId>
   <gameFeature>BUY-8</gameFeature>
   <transferTime>20120720T230043</transferTime>
   <specialGame>
      <type>Free</type>
```

```
      <count>10</count>
      <sequence>1</sequence>
   </specialGame>
   <ref_ticket_ids>[120001, 120002, 120003]</ref_ticket_ids>
</TransferRequest>
```

JSON TransferResponse：

```
{
      "transferId": "0ab9bdca06c14811b24653468e609838",
      "merchantCode": "SPADE",
      "merchantTxId": "20130813014319279367",
      "acctId": "TESTPLAYER1",
      "balance": 1050,
      "msg": "success",
      "code": 0,
      "serialNo": "20120722231413699735"
}
```

XML TransferResponse：

```
<TransferResponse>
   <serialNo>20120722231413699735</serialNo>
   <code>0</code>
   <msg>success</msg>
   <merchantCode>SPADE</merchantCode>
   <transferId>0ab9bdca06c14811b24653468e60988</transferId>
   <merchantTxId>20130813014319279367</merchantTxId>
   <acctId>TESTPLAYER1</acctId>
   <balance>1050</balance>
</TransferResponse>
```

4.3.3 Bonus Transfer

Usage：

1. Apply on promotion activities. During promotion activities, if players triggered bonus, system would send type= 7 to Merchant.
2. When Game Provider send bonus transfer request (type=7) to merchant， merchant must able to respond and increase the player balance.
3. Merchant must develop this API to participate planned promotion activities in near future.

Interface Name：transfer

API Provider：Merchant

API Requestor：Game Provider

Request Message Definition for `TransferRequest`：

Name	Data Type	Mandatory	Example	Description
transferId	Varchar(50)	Yes	a3b0c9dd1ab041	Unique transfer Id
acctId	Varchar(50)	Yes	TESTPLAYER1	Player id
currency	Char(3)	Yes	USD	Currency code
amount	Decimal(20,9)	Yes	1000	Transfer amount (must greater than or equal to 0)
type	Int	Yes	7	7 = Bonus
channel	Varchar(10)	Yes	Web/Mobile, APP-i/APP-A/PC	Ticket for mobile or web
gameCode	Varchar(10)	Yes	B-SG02	Game Code
ticketId	Varchar(20)	Yes	641482277	
referenceId	Varchar(50)	No		Don’t have reference Id due to don’t have placebet
roundId	Varchar(20)	No	432	Promotion’s Id
siteId	Varchar(20)	No	DEMO1	Client’s site Id

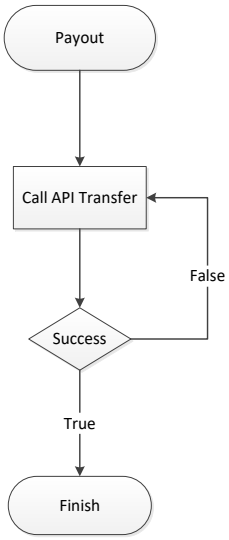
Important Note:

1. Bonus don’t have reference Id, due to don’t placebet.
2. RoundId and siteId will be enabled upon request

Response message definition for `TransferResponse`：

Name	Data Type	Mandatory	Example	Description
transferId	Varchar(50)	Yes	a3b0c9dd1ab041	Unique transfer Id
merchantTxId	Varchar(20)	No	498036704	Merchant transaction id
acctId	Varchar(50)	Yes	TESTPLAYER1	Player ID
balance	Decimal(20,4)	Yes	1000	Player balance after transfer

Bonus Flow:



Transfer Request and response messages example:

JSON TransferRequest：

```
{
      "acctId": "TESTPLAYER1",
      "transferId": "a3b0c9dd1ab041",
      "currency": "USD",
      "amount": 10,
      "type": 7,
      "ticketId": "234950357",
      "channel": "Web",
      "gameCode": "B-FS02",
      "merchantCode": "SPADE",
      "serialNo": "20120722231413699735",
      "referenceId": ""
}
```

XML TransferRequest：

```
<TransferRequest>
   <serialNo>20120722231413699735</serialNo>
   <merchantCode>SPADE</merchantCode>
   <transferId>a3b0c9dd1ab041</transferId>
   <acctId>TESTPLAYER1</acctId>
   <currency>USD</currency>
   <amount>10</amount>
   <type>7</type>
   <ticketId>234950357</ticketId>
   <channel>Web</channel>
   <gameCode>B-FS02</gameCode>
   <referenceId />
</TransferRequest>
```

JSON TransferResponse：

```
{
      "transferId": "a3b0c9dd1ab041",
      "merchantCode": "SPADE",
      "merchantTxId": "20130813014319279367",
      "acctId": "TESTPLAYER1",
      "balance": 1050,
      "msg": "success",
      "code": 0,
      "serialNo": "20120722231413699735"
}
```

XML TransferResponse：

```
<TransferResponse>
   <serialNo>20120722231413699735</serialNo>
   <code>0</code>
   <msg>success</msg>
   <merchantCode>SPADE</merchantCode>
   <transferId>a3b0c9dd1ab041</transferId>
   <merchantTxId>20130813014319279367</merchantTxId>
   <acctId>TESTPLAYER1</acctId>
   <balance>1050</balance>
</TransferResponse>
```

4.3.4 Query transfer history

Usage：Search for transfer history

Interface Name：transferHistory

API Provider：Game Provider

API Requestor：Merchant

Request Message definition `TransferHistoryRequest`：

Name	Data Type	Mandatory	Example	Description
beginDate	Char(15)	Yes	20120720T230043	Start date range to search. Format yyyymmdd'T'24hhmmss
endDate	Char(15)	Yes	20120722T225043	End date range to search. Format yyyymmdd'T'24hhmmss
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code
pageIndex	Integer	Yes	1	Page number
transferId	Varchar(50)	No	a3b0c9dd1ab041	Unique transfer Id

Response message definition `TransferHistoryResponse`：

Name	Data Type	Mandatory	Example	Description
resultCount	Integer	Yes	100	Total record count
pageCount	Integer	Yes	1	Total page count
list	TransferInfo[]	Yes		TransferInfo Array

TransferInfo Definition `TransferInfo`：

Name	Data Type	Mandatory	Example	Description
transferId	Varchar(50)	Yes	a3b0c9dd1ab041	Unique transfer Id
acctId	Varchar(50)	Yes	11	Player ID
ticketId	Varchar(20)	Yes	641482277	
transferTime	Char(15)	Yes	20120722T225043	Transfer Time
merchantTxid	Varchar(20)	Yes	D141215030051fmowg	Merchant Transfer Id
type	Int	Yes	1	1 = place bet 2 = cancel bet 4 = payout 7 = Bonus
amount	Decimal(20,4)	Yes	1000	Transfer Amount
status	Int	Yes	0	Transfer status
channel	Varchar(10)	Yes	Web/Mobile, APP-i/APP-A/PC	Ticket for mobile or web

TransferHistory Request and response messages example:

JSON TransferHistoryRequest：

```
{
      "beginDate": "20120721T175139",
      "endDate": "20120723T174139",
      "pageIndex": 1,
      "merchantCode": "SPADE",
      "serialNo": "20120723175139440637",
      "transferId": "667706462"
}
```

XML TransferHistoryRequest：

```
<TransferHistoryRequest>
   <serialNo>20120723175139440637</serialNo>
   <merchantCode>SPADE</merchantCode>
   <beginDate>20120721T175139</beginDate>
   <endDate>20120723T174139</endDate>
   <pageIndex>1</pageIndex>
   <transferId>667706462</transferId>
</TransferHistoryRequest>
```

JSON TransferHistoryResponse：

```
{
      "list": [
            {
                  "transferId": "667706462",
                  "acctId": "TESTPLAYER1",
                  "ticketId": "749084635",
                  "type": 1,
                  "amount": 20,
                  "transferTime": "20130908T151100",
                  "merchantTxId": "D141215030051fmowg",
                  "status": 0,
                  "channel": "Web"
            }
      ],
      "resultCount": 1,
      "pageCount": 1,
      "merchantCode": "SPADE",
      "code": 0,
      "msg": "success",
      "serialNo": "20120726214956563472"
```

```
}
```

XML TransferHistoryResponse：

```
<TransferHistoryResponse>
   <serialNo>20120726214956563472</serialNo>
   <code>0</code>
   <msg>success</msg>
   <merchantCode>SPADE</merchantCode>
   <resultCount>1</resultCount>
   <pageCount>1</pageCount>
   <list>
      <TransferInfo>
         <transferId>667706462</transferId>
         <acctId>TESTPLAYER1</acctId>
         <ticketId>749084635</ticketId>
         <type>1</type>
         <amount>20</amount>
         <transferTime>20130908T151100</transferTime>
         <merchantTxId>D141215030051fmowg</merchantTxId>
         <status>0</status>
         <channel>Web</channel>
      </TransferInfo>
   </list>
</TransferHistoryResponse>
```

4.4 Common API

4.4.1 Query player betting history

Usage：Search for bet history

Interface Name：getBetHistory

API Provider：Game Provider
API Requestor：Merchant

Request Message Definition for `GetBetHistoryRequest`：

Name	Data Type	Mandatory	Example	Description
beginDate	Char(15)	Yes	20120720T230043	Start date range to search. Format must be in yyyymmdd'T'24hhmmss
endDate	Char(15)	Yes	20120722T225043	End date range to search. Format must be in yyyymmdd'T'24hhmmss
pageIndex	Integer	Yes	1	Page number
serialNo	Varchar(50)	Yes	20120722231413595602	Generated by merchant to indicate message sequence.
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code

Response message definition for `GetBetHistoryResponse`：

Name	Data Type	Mandatory	Example	Description
resultCount	Integer	Yes	100	Total record count
pageCount	Integer	Yes	1	Total page count
list	BetInfo[]	Yes		BetInfo Array

`BetInfo` definition：

Name	Data Type	Mandatory	Example	Description
ticketId	Varchar(20)	Yes	641482277	Ticket number.
acctId	Varchar(50)	Yes	TESTPLAYER1	Player's unique Id
ticketTime	Char(15)	Yes	20120722T225417	Date time when ticket is bet
categoryId	Varchar(10)	Yes	SM or BC or AD or BN or FH	Game category SM=Slot; BC=Table; AD=Arcade; BN=Promotion; FH=Fishing
gameCode	Varchar(10)	Yes	S-DG02	Game Code
currency	Char(3)	Yes	USD	Currency ISO code
betAmount	Decimal(18,6)	Yes	20	Bet Amount
result	Varchar(1024)	No	8,8,5,10,10,7,7	Result
winLoss	Decimal(18,6)	Yes	-20	Member win loss.
jackpotAmount	Decimal(18,6)	Yes	0	
betIp	Varchar(50)	Yes	8.8.8.8	Bettor's IP Address
luckyDrawId	Long	No	10	Promotion ID
roundId	Integer	Yes	1823	Game Round ID
sequence	Integer	Yes	0,1,2,3,4	0 = No jackpot win

channel	Varchar(10)	Yes	Mobile/Web	Ticket for mobile or web
Balance	Decimal(18,6)	Yes	234.000000	Previous Balance
jpWin	Decimal(18,6)	Yes	-100.000000	Jackpot Win
referenceId	Varchar(50)	No	a8ab9cc8f1a277de	Reference to transfer id
gameFeature	Varchar(50)	No	BUY-20	Only response when player trigger Buy Feature

Note :

Ticket is based on game payout datetime. Recommend retrieve interval is within 1-5 minutes and each ticket retrieve after 3 minutes of payout time. Maximum retrieve interval is 2 hours.

It has concurrent call limit for this query, each query will only enable SINGLE request. The next request will only enable after current request has completed. Otherwise, it will return code: 101 USER_CALL_LIMITED

GetBetHistory (Request and response messages example):

JSON GetBetHistoryRequest：

```
{
      "beginDate": "20120721T175139",
      "endDate": "20120723T174139",
      "pageIndex": 1,
      "merchantCode": "SPADE",
      "serialNo": "20120723175139440637"
}
```

XML GetBetHistoryRequest：

```
<GetBetHistoryRequest>
      <serialNo>20120723175139440637</serialNo>
      <merchantCode>SPADE</merchantCode>
      <beginDate>20120721T175139</beginDate>
      <endDate>20120723T174139</endDate>
      <pageIndex>1</pageIndex>
</GetBetHistoryRequest>
```

JSON GetBetHistoryResponse：

```
{
      "resultCount": 1,
      "list": [
            {
                  "ticketId": 633161396,
                  "acctId": "TESTPLAYER1",
                  "categoryId": "SM",
                  "gameCode": "SS-GD02",
                  "ticketTime": "20130908T161044",
```

```
                "betIp": "8.8.8.8",
                "betAmount": 20,
                "winLoss": 30,
                "result": "23412",
                "jackpotAmount": 1.5,
                "luckyDrawId": 0,
                "roundId": 1879,
                "sequence": 0,
                "channel": "web",
                "balance": 234.0,
                "jpWin": 0.0,
                "referenceId": "arwwew232ghfg4dfsdfsdwr2",
                "gameFeature": "BUY-20"
            }
        ],
        "merchantCode": "SPADE",
        "msg": "success",
        "code": 0,
        "serialNo": "20120723175139440637"
}
```


XML GetBetHistoryResponse：


```
<GetBetHistoryResponse>
   <serialNo>20120723175139440637</serialNo>
   <code>0</code>
   <msg>success</msg>
   <merchantCode>SPADE</merchantCode>
   <resultCount>1</resultCount>
   <list>
      <BetInfo>
         <ticketId>633161396</ticketId>
         <acctId>Test10001</acctId>
         <categoryId>SM</categoryId>
         <gameCode>SS-GD02</gameCode>
         <ticketTime>20130908T161044</ticketTime>
         <betIp>8.8.8.8</betIp>
         <betAmount>20</betAmount>
         <winLoss>30</winLoss>
         <jackpotAmount>1.5</jackpotAmount>
         <result>23412</result>
         <luckyDrawId>0</luckyDrawId>
         <sequence>0</sequence>
         <roundId>1387</roundId>
         <channel>Web</channel>
         <balance>234.000000</balance>
         <jpWin>0.0</jpWin>
         <referenceId>arwwew232ghfg4dfsdfsdwr2</referenceId>
        <gameFeature>BUY-20</gameFeature>
```

```
      </BetInfo>
   </list>
</GetBetHistoryResponse>
```

4.4.2 Query merchant game info

Usage：Query merchant game info

Interface Name：getGames

API Provider：Game Provider

API Requestor：Merchant

Request message definition for `MerchantGamesRequest`：

Name	Data Type	Mandatory	Example	Description
serialNo	Varchar(50)	Yes	20120722231413595602	Generated by merchant to indicate message sequence.
merchantCode	Varchar(50)	Yes	SPADE	Merchant Code
currency	Char(3)	No	USD	Currency Code
language	Varchar(10)	No	th TH (Appendix 2)	Language code

Response message definition for `MerchantGamesResponse`：

Name	Data Type	Mandatory	Example	Description
games	GameInfo[]	yes		Game Info class

GameInfo Definition `GameInfo`：

Name	Data Type	Mandatory	Example	Description
gameCode	Varchar(10)	yes	SS-GD02	Game code
gameName	Varchar(30)	yes	DerbyNight	Game name
jackpot	boolean	yes	True	Is jackpot
thumbnail	Varchar(100)	yes	/images/aa.jpg	Game picture
screenshot	Varchar(100)	yes	/images/bb.jpg	Game screen shot
jackpotCode	Varchar(50)	no	Holy	Only for Jackpot Game
jackpotName	Varchar(50)	no	Holy Jackpot	Only for Jackpot Game

Note:

1. The full path of the picture is http://$host/thumbnail/aa.jpg, http://$host/screenshot/bb.jpg. $host is the host part of the API URL.
   For example, if the API URL is “http://demoapi.egame.spadegaming.com/api”, then the picture full path is “http://demoapi.egame.spadegaming.com/thumbnail/aa.jpg”

2. The API is used for background job. For our API is server to server, the pictures cannot access for public IP. We recommend run the job every day or every week, to download the pictures and display in website.

getGames(Request and response messages example):

JSON getGamesRequest:

```
{
      "merchantCode": "SPADE",
      "serialNo": "20130502191551906534"
}
```

XML getGamesRequest:

```
<getGamesRequest>
   <serialNo>20130502191551906534</serialNo>
   <merchantCode>SPADE</merchantCode>
</getGamesRequest>
```

JSON getGamesResponse:

```
{
      "games": [
            {
                  "gameCode": "S-DG02",
                  "gameName": "DerbyNight",
                  "jackpot": true,
                  "thumbnail": "/thumbnail/S-DG02.jpg",
                  "screenshot": "/screenshoot/S-DG02.jpg",
                  "jackpotCode ": "Holy",
                  "jackpotName ": "Holy Jackpot"
            }
      ],
      "merchantCode": "SPADE",
      "code": 0,
      "msg": "success",
      "serialNo": "20130502191551906534"
}
```

XML getGamesResponse:

```
<getGamesResponse>
   <games>
      <gameCode>S-DG02</gameCode>
      <gameName>DerbyNight</gameName>
      <jackpot>true</jackpot>
      <thumbnail>/thumbnail/S-DG02.jpg</thumbnail>
      <screenshot>/screenshoot/S-DG02.jpg</screenshot>
      <mthumbnail_>http://xxxx.com/aa.jpg</mthumbnail_>
      <jackpotCode_>Holy</jackpotCode_>
      <jackpotName_>Holy Jackpot</jackpotName_>
   </games>
   <merchantCode>SPADE</merchantCode>
   <code>0</code>
   <msg>success</msg>
   <serialNo>20130502191551906534</serialNo>
</getGamesResponse>
```

4.4.3 Force Logout for online members

Usage：Kick selected user or all users out of E-games

Interface Name：kickAcct
API Provider：Game Provider
API Requestor：Merchant

Request Message Definition for `KickAcctRequest`：

Name	Data Type	Mandatory	Example	Description
gameCode	Varchar(10)	No	S-DG02	Game code
acctId	Varchar(50)	No	TESTPLAYER1	Player ID, Leave empty to force all “online” users to log out of E-games/Lottery.
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code
serialNo	Varchar(50)	Yes	20120722231413595602	Generated by merchant to indicate message sequence.

Note:

1. Game code = Kick out player by game
2. Acct id = Kick out certain player.
3. Game code + acct id = Kick out certain player from certain game.

Response message definition for `KickAcctResponse`：

Name	Data Type	Mandatory	Example	Description
serialNo	Varchar(50)	Yes	20120722224255982841	Generated by merchant to indicate message sequence.
code	Integer	Yes	0	Processed result status.
msg	Varchar(128)	Yes	success	Processed result description.
merchantCode	Varchar(10)	Yes	SPADE	Unique code that identifies the Merchant.

KickAcct(Request and response messages example):

JSON KickAcctRequest:

```
{
      "gameCode": "S-DG02",
      "acctId": "TESTPLAYER1",
      "merchantCode": "SPADE",
      "serialNo": "20130430164644935780"
}
```

XML KickAcctRequest:

```
<kickAcctRequest>
  <gameCode>S-DG02</gameCode>
  <serialNo>20130430164644935780</serialNo>
  <merchantCode>SPADE</merchantCode>
  <acctId>TESTPLAYER1</acctId>
</kickAcctRequest>
```

JSON KickAcctResponse:

```
{
      "merchantCode": "SPADE",
      "code": 0,
      "msg": "success",
      "serialNo": "20130430164644935780"
}
```

XML KickAcctResponse:

```
<kickAcctResponse>
   <serialNo>20130430164644935780</serialNo>
   <code>0</code>
   <msg>success</msg>
   <merchantCode>SPADE</merchantCode>
</kickAcctResponse>
```

4.4.4 Query Exist of player Acct

Usage：Merchant wishes to check player Acct whether exist or not.

Interface Name：checkAcct

API Provider：Game Provider

API Requestor：Merchant

Request Message Definition for `CheckAcctRequest`：

Name	Data Type	Mandatory	Example	Description
acctId	Varchar(50)	Yes	TESTPLAYER1	Unique player ID
includeType	Boolean	No	True	Confirming is “youtuber” player or not
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code
serialNo	Varchar(50)	Yes	20120722231413595602	generated by "requestor"

Response message definition for `CheckAcctResponse`：

Name	Data Type	Mandatory	Example	Description
serialNo	Varchar(50)	Yes	20120722224255982841	Same as request’s serialNo
acctType	Integer	No	0	The definition of code 0 : normal 1 : youtuber
code	Integer	Yes	0	Processed result status.
msg	Varchar(128)	Yes	success	Processed result description.
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code

checkAcct Request and response messages example:

JSON checkAcct Request：

```
{
      "acctId": "TESTPLAYER1",
      "merchantCode": "SPADE",
      "includeType": true,
      "serialNo": "20180722231413699735"
}
```

XML checkAcct Request：

```
<checkAcctRequest>
   <acctId>TESTPLAYER1</acctId>
   <includeType>true</includeType>
   <merchantCode>SPADE</merchantCode>
```

```
   <serialNo>20180722231413699735</serialNo>
</checkAcctRequest>
```

JSON checkAcct Response：

```
{
      "serialNo":"20180722231413699735",
      "code":0,
      "acctType":0,
      "msg":"Success",
      "merchantCode":"SPADE"
}
```

XML checkAcct Response：

```
<checkAcctResponse>
   <serialNo>20180722231413699735</serialNo>
   <code>0</code>
   <acctType>0</acctType>
   <msg>Success</msg>
   <merchantCode>SPADE</merchantCode>
</checkAcctResponse>
```

4.4.5 Query player daily summary by game

Usage：Search for player daily summary by each game

Interface Name：playerDailySumByGame

API Provider：Game Provider
API Requestor：Merchant

Request Message Definition for `playerDailySumByGameRequest`：

Name	Data Type	Mandatory	Example	Description
beginDate	Char(15)	Yes	20120721T000000	Start date range to search. Format must be in yyyymmdd'T'24hhmmss
endDate	Char(15)	Yes	20120721T010000	End date range to search. Format must be in yyyymmdd'T'24hhmmss
pageIndex	Integer	Yes	1	Page number
serialNo	Varchar(50)	Yes	20120722231413595602	Generated by merchant to indicate message sequence.
merchantCode	Varchar(50)	Yes	SPADE	Merchant Code
acctId	Varchar(50)	No	TESTPLAYER1	Player's unique Id

Response message definition for `playerDailySumByGameResponse`：

Name	Data Type	Mandatory	Example	Description
resultCount	Integer	Yes	100	Total record count
pageCount	Integer	Yes	1	Total page count
list	PlayerBetInfo[]	Yes		PlayerBetInfo Array

`PlayerBetInfo` definition：

Name	Data Type	Mandatory	Example	Description
acctId	Varchar(50)	Yes	TESTPLAYER1	Player ID
list	BetInfo[]	Yes		BetInfo Array

`BetInfo` definition：

Name	Data Type	Mandatory	Example	Description
bets	Integer	Yes	1	Bet Count
betAmount	Decimal(20,4)	Yes	20	Bet Amount
winLoss	Decimal(18,6)	Yes	-20	Member win loss.
jackpotAmount	Decimal(18,6)	Yes	0	
jpWin	Decimal(18,6)	Yes	-100.000000	Jackpot Win
categoryId	Varchar(10)	Yes	SM or TB or AD or BN	Game Category
gameCode	Varchar(10)	Yes	S-GD02	Game Code

Notes :

1. Maximum retrieve interval is 1 day.
2. Retrieve data datetime must be 12 hours ago from current datetime.
3. Data query by hour

playerDailySumByGame (Request and response messages example):

JSON playerDailySumByGameRequest：

```
{
      "beginDate": "20120721T000000",
      "endDate": "20120721T010000",
      "pageIndex": 1,
      "merchantCode": "SPADE",
      "serialNo": "20120723175139440637"
}
```

XML playerDailySumByGameRequest：

```
<PlayerDailySumByGameRequest>
   <serialNo>20120723175139440637</serialNo>
   <merchantCode>SPADE</merchantCode>
   <beginDate>20120721T000000</beginDate>
   <endDate>20120721T010000</endDate>
   <pageIndex>1</pageIndex>
</PlayerDailySumByGameRequest>
```

JSON playerDailySumByGameResponse：


```
{
      "resultCount": 2,
      "pageCount": 1,
      "list": [
            {
                  "acctId": "Test10001",
                  "list": [
                        {
                              "bets": 34,
                              "betAmount": 20,
                              "winLoss": 30,
                              "jackpotAmount": 1.5,
                              "jpWin": -100.0,
                              "categoryId": "SM",
                              "gameCode": "S-GD02"
                        }
                  ]
            },
            {
                  "acctId": "Test10002",
                  "list": [
                        {
                              "bets": 34,
                              "betAmount": 20,
                              "winLoss": 30,
                              "jackpotAmount": 1.5,
                              "jpWin": -100.0,
                              "categoryId": "SM",
                              "gameCode": "S-GD02"
                        }
                  ]
            }
      ],
      "merchantCode": "SPADE",
      "msg": "success",
      "code": 0,
      "serialNo": "20120723175139440637"
}
```


XML playerDailySumByGameResponse：


```
<PlayerDailySumByGameResponse>
   <serialNo>20120723175139440637</serialNo>
   <code>0</code>
   <msg>success</msg>
   <merchantCode>SPADE</merchantCode>
```

```
   <resultCount>1</resultCount>
   <list>
      <PlayerBetInfo>
         <acctId>Test10001</acctId>
         <list>
            <BetInfo>
               <bets>20</bets>
               <betAmount>20</betAmount>
               <winLoss>30</winLoss>
               <jackpotAmount>0</jackpotAmount>
               <jpWin>-100.000000</jpWin>
               <categoryId>SM</categoryId>
               <gameCode>S-GD02</gameCode>
            </BetInfo>
         </list>
      </PlayerBetInfo>
   </list>
</PlayerDailySumByGameResponse>
```

4.4.6 Query player ticket detail by page

Usage: Player view ticket details by clicking *View ticket / Transfer details* button found in merchant website.

Interface Name： getTicketLog

API Provider：Game Provider
API Requestor：Merchant

Request Message Definition for `getTicketLogRequest`：

Name	Data Type	Mandatory	Example	Description
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code
acctId	Varchar(50)	Yes	TESTPLAYER1	Unique player ID
ticketId	Varchar(20)	Yes	641482277	Ticket number.
action	Varchar(50)	Yes	ticketLog	view the ticket detail by ticketId
serialNo	Varchar(80)	Yes	201207222314135956 02	Generated by merchant to indicate message sequence.
language	Varchar(10)	No	en_US	Language code. Refer Appendix 2

Response message definition for `getTicketLogResponse`：

Name	Data Type	Mandatory	Example	Description
token	Varchar(80)	Yes	a3b0c9dd1ab041	Unique token for merchant
serialNo	Varchar(50)	Yes	498036704	Merchant transaction id
ticketUrl	Varchar(200)	Yes		Game url
msg	Varchar(50)	Yes	success	
code	Int	Yes	0	Response Code

Note:

1. We advise to use GUID (UUID), or the date+time+random string as the SerialNo

getTicketLog (Request and response messages example):

JSON getTicketLogRequest：

```
{
      "merchantCode": " SPADE",
      "acctId": "TESTPLAYER1",
      "ticketId": "641482277",
      "language": "en_US",
      "serialNo": "20120722231413595602",
      "action": "ticketLog"
}
```

XML getTicketLogRequest：

```
<getTicketLogRequest>
      <merchantCode>SPADE</merchantCode>
      <acctId>TESTPLAYER1</acctId>
      <ticketId>641482277</ticketId>
      <language>en_US</language>
      <action>ticketLog</action>
</getTicketLogRequest>
```

JSON getTicketLogResponse：

```
{
      "ticketUrl":"
https://example.domain.com/?ticketId=641482277&acctId=
TESTPLAYER1&action=ticketLog&token=928d77dbc0aec1e01a6636210fdbf63d
&language=en_US&merchantCode=SPADE"
      "msg": "success",
      "code": 0,
      "serialNo": "20120722231413595602"
}
```

XML getTicketLogResponse:

```
<getTicketLogResponse>
<ticketUrl>https://example.domain.com/?ticketId=641482277&amp
;acctId=
TESTPLAYER1&amp;action=ticketLog&amp;token=928d77dbc0aec1e01a
6636210fdbf63d&amp;language=en_US&amp;merchantCode=SPADE</tic
```

```
ketUrl>
   <msg>success</msg>
   <code>0</code>
   <serialNo>20120722231413595602</serialNo>
</getTicketLogResponse>
```

4.4.7 Query player transfer detail by page

Usage: Player view ticket details by clicking *View TransferId details* button found in merchant website.

Note: Only available for common wallet

Interface Name：getTicketLog

API Provider：Game Provider
API Requestor：Merchant

Request Message Definition for `getTicketLogRequest`:

Name	Data Type	Mandatory	Example	Description
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code
acctId	Varchar(50)	Yes	TESTPLAYER1	Unique player ID
transferId	Varchar(20)	No	20120722231413595602	Unique transfer Id (placebet or bonus transferId)
action	Varchar(50)	Yes	allTransferTicketLog	view the ticket detail by placebet or bonus transfer Id
serialNo	Varchar(80)	Yes	20120722231413595602	Generated by merchant to indicate message sequence.
language	Varchar(10)	No	en_US	Language code. Refer Appendix 2

Response message definition for `getTicketLogResponse`:

Name	Data Type	Mandatory	Example	Description
serialNo	Varchar(50)	Yes	498036704	Merchant transaction id
ticketUrl	Varchar(200)	Yes		Game url
msg	Varchar(50)	Yes	success	
code	Int	Yes	0	Response Code

Note:

1. We advise to use GUID (UUID), or the date+time+random string as the SerialNo

getTicketLog (Request and response messages example):

JSON getTicketLogRequest：

```
{
      "merchantCode": " SPADE",
      "acctId": "TESTPLAYER1",
      "transferId": "20120722231413595602",
      "language": "en_US",
      "serialNo": "20120722231413595602",
      "action": "allTransferTicketLog"
}
```

XML getTicketLogRequest：

```
<getTicketLogRequest>
      <merchantCode>SPADE</merchantCode>
      <acctId>TESTPLAYER1</acctId>
      <ticketId>641482277</ticketId>
      <language>en_US</language>
      <action>allTransferTicketLog</action>
</getTicketLogRequest>
```

JSON getTicketLogResponse：

```
{
      "ticketUrl": "
https://example.domain.com/?transferId=20120722231413595602&acctId=
TESTPLAYER1&action=allTransferticketlog&token=928d77dbc0aec1e01a663
6210fdbf63d&language=en_US&merchantCode=SPADE",
      "msg": "success",
      "code": 0,
      "serialNo": "20120722231413595602"
}
```

XML getTicketLogResponse:

```
<getTicketLogResponse>
<ticketUrl>https://example.domain.com/?transferId=20120722231413595
602&amp;acctId=
TESTPLAYER1&amp;action=allTransferTicketLog&amp;token=928d77dbc0aec
1e01a6636210fdbf63d&amp;language=en_US&amp;merchantCode=SPADE</tick
```

```
etUrl>
   <serialNo>201207222314135956O2</serialNo>
   <code>0</code>
   <msg>success</msg>
</getTicketLogResponse>
```

4.4.8 set Youtuber

Usage： When merchant wants to set youtuber feature for a particular member.

Interface Name：setYoutuber

API Provider：Game Provider
API Requestor：Merchant

Request Message Definition for `setYoutuberRequest`：

Name	Data Type	Mandatory	Example	Description
acctId	Varchar(50)	Yes	TESTPLAYER1	Unique player ID
currency	Char(3)	Yes	USD	Currency Code
youtuber	Boolean	Yes	True	True: Youtuber False: Normal
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code
serialNo	Varchar(50)	Yes	20120722231413595602	generated by "requestor"

Response message definition for `setYoutuberResponse`：

Name	Data Type	Example	Description
serialNo	Varchar(50)	20120722224255982841	Generated by merchant to indicate message sequence.
code	Integer	0	Processed result status.
msg	Varchar(128)	success	Processed result description.
merchantCode	Varchar(10)	SPADE	Unique code that identifies the Merchant.

setYoutuber (Request and response messages example):

JSON setYoutuberRequest：

```
{
      "acctId": "TESTPLAYER1",
      "currency": "USD",
      "youtuber": true,
      "merchantCode": "SPADE",
      "serialNo": "20120722231413699735"
}
```

XML setYoutuberRequest：

```
<setYoutuberRequest>
   <acctId>TESTPLAYER1</acctId>
   <currency>USD</currency>
   <merchantCode>SPADE</merchantCode>
   <serialNo>20120722231413699735</serialNo>
```

```
    <youtuber>true</youtuber>
</setYoutuberRequest>
```

JSON setYoutuberResponse：


```
{
      "merchantCode": "SPADE",
      "msg": "success",
      "code": 0,
      "serialNo": "20120722231413699735"
}
```


XML setYoutuberResponse：


```
<setYoutuber>
   <code>0</code>
   <merchantCode>SPADE</merchantCode>
   <msg>success</msg>
   <serialNo>20120722231413699735</serialNo>
</setYoutuber>
```

4.4.9 Query Jackpot pool

Usage：Merchant query the jackpot pool amount and display on merchant website

Interface Name：jackpotPool

API Provider：Game Provider

API Requestor：Merchant

Request Message Definition for `JackpotPoolRequest`：

Name	Data Type	Mandatory	Example	Description
currency	Char(3)	Yes	USD	Currency ISO code

** If currency is empty, then all jackpot for each currency will listed

Response message definition for `JackpotPoolResponse`：

Name	Data Type	Mandatory	Example	Description
list	JackpotPoolInfo[]	Yes		JackpotPoolInfo type

JackpotPoolInfo type definition：

Name	Data Type	Mandatory	Example	Description
code	Varchar(20)	Yes	Grand	Jackpot Code
name	Varchar(50)	No	Grand Jackpot	Jackpot Name
amount	Decimal(20,2)	Yes	3844.08	Jackpot Pool amount

JackpotPool (Request and response messages example):

JSON JackpotPoolRequest：

```
{
      "currency": "USD",
      "merchantCode": "SPADE",
      "serialNo": "20120726214956563472"
}
```

XML JackpotPoolRequest：

```
<JackpotPoolRequest>
  <serialNo>20120726214956563472</serialNo>
  <merchantCode>SPADE</merchantCode>
  <currency>USD</currency>
</JackpotPoolRequest>
```

JSON JackpotPoolResponse：


```
{
      "list": [
            {
                  "code": "Grand",
                  "name": "Grand Jackpot",
                  "amount": 3844.08,
                  "currency": "USD"
            }
      ],
      "status": 0,
      "merchantCode": "SPADE",
      "msg": "success",
      "code": 0,
      "serialNo": "20120726214956563472"
}
```


XML JackpotPoolResponse：


```
<JackpotPoolResponse>
   <serialNo>20120802152140143938</serialNo>
   <code>0</code>
   <msg>0</msg>
   <merchantCode>SPADE</merchantCode>
   <resultCount>1</resultCount>
   <pageCount>1</pageCount>
   <list>
      <JackpotPoolInfo>
         <code>Grand</code>
         <name>Grand Jackpot</name>
         <amount>3844.08</amount>
         <currency>USD</currency>
      </JackpotPoolInfo>
   </list>
</JackpotPoolResponse>
```

4.4.10 Query Jackpot pool of merchant currency

Usage：Merchant query the jackpot pool amount and display on merchant website

Interface Name：jackpotPoolAll

API Provider：Game Provider

API Requestor：Merchant

Request Message Definition for `JackpotPoolRequest`：

Name	Data Type	Mandatory	Example	Description
merchantCode	Varchar(50)	Yes	SPADE	Merchant Code
serialNo	Varchar(50)	Yes	20120722231413595602	Generated by merchant to indicate message sequence.

Response message definition for `JackpotPoolResponse`：

Name	Data Type	Mandatory	Example	Description
list		Yes		JackpotPoolInfo type

JackpotPoolInfo type definition：

Name	Data Type	Mandatory	Example	Description
code	Varchar(20)	Yes	Grand	Jackpot Code
amoutMap	Varchar(50)	Yes	"CNY":103418.588499	jackpotAmount

JackpotPool (Request and response messages example):

JSON JackpotPoolAllRequest：

```
{
      "merchantCode": "SPADE",
      "serialNo": "20120726214956563472"
}
```

XML JackpotPoolAllRequest：

```
<JackpotPoolAllRequest>
  <serialNo>20120726214956563472</serialNo>
  <merchantCode>SPADE</merchantCode>
</JackpotPoolAllRequest>
```

JSON JackpotPoolAllResponse：


```
{
	"list": [
		{
			"code": "Grand",
			"amountMap":
			{
				"CNY":103418.588499,
				"IDR":3991172.548579,
				"HKD":2175666.803645
			}
		},
		{
			"code":"GJ",
			"amountMap":
			{
				"CNY":16494.54591,
				"IDR":30333469.927946,
				"HKD":1944453.200509
			}
		}
	]
	"status": 0,
	"merchantCode": "SPADE",
	"msg": "success",
	"code": 0,
	"serialNo": "20120726214956563472"
}
```

XML JackpotPoolResponse：

```
<JackpotPoolAllResponse>
   <serialNo>20120802152140143938</serialNo>
   <code>0</code>
   <msg>0</msg>
   <merchantCode>SPADE</merchantCode>
   <resultCount>1</resultCount>
   <pageCount>1</pageCount>
   <list>
           <code>Grand</code>
       <amountMap>
                <CNY>103418.588499</CNY>
                <IDR>3991172.548579</IDR>
                <HKD>2175666.803645</HKD>
           </amountMap>
           <code>GJ</code>
       <amountMap>
                <CNY>16494.54591</CNY>
                <IDR>30333469.927946</IDR>
                <HKD>1944453.200509</HKD>
           </amountMap>

   </list>
</JackpotPoolAllResponse>
```

4.4.11 Query the Member List Status

Usage：Search the online/offline member status

Interface Name：getMemberList

API Provider：Game Provider

API Requestor：Merchant

Request message definition for `getMemberListRequest`:

Name	Data Type	Mandatory	Example	Description
serialNo	Varchar(50)	Yes	20120722231413595602	generated by “requestor"
merchantCode	Varchar(10)	Yes	SPADE	Merchant Code
currency	Char(3)	No	USD	Currency Code
pageIndex	Integer	Yes	1	Page number
Online	Boolean	Yes	true	true = Online Member false =offline Member
hasBalance	Boolean	No	true	true = has balance false = no balance

Response message definition for `getMemberListResponse`:

Name	Data Type	Mandatory	Example	Description
list	AcctInfo []	yes		AcctInfo

AcctInfo type definition：

Name	Data Type	Mandatory	Example	Description
acctId	Varchar(50)	Yes	TESTPLAYER1	Unique player ID
currency	Char(3)	Yes	USD	Currency Code
balance	Decimal(20,4)	No	1000	Current player balance

getMemberList(Request and response messages example):

JSON getMemberListRequest：

```
{
      "merchantCode": "SPADE",
      "serialNo": "20130502191551906534",
      "pageIndex": 1,
      "online":"true",
      "hasBalance":"true",
      "currency": USD
}
```

XML getMemberListRequest:

```
<getMemberList>
  <serialNo>20130502191551906534</serialNo>
  <merchantCode>SPADE</merchantCode>
  <currency>USD</currency>
  <online>true</online>
  <hasBalance>true</hasBalance>
```

```
</getMemberList>
```

JSON getMemberListResponse：

```
{
      "list": [{
            "currency": "USD",
            "acctId": "TESTPLAYER1",
            "balance": 10

      }],
      "merchantCode": "SPADE",
      "code": 0,
      "msg": "success",
      "serialNo": "20130502191551906534"
}
```

XML getMemberListResponse:

```
<getMemberListResponse>
  <serialNo>20130502191551906534</serialNo>
  <code>0</code>
  <msg>success</msg>
  <merchantCode>SPADE</merchantCode>
  <list>
      <currency>USD</currency>
       <acctId>TESTPLAYER1</acctId>
       <balance>10</balance>
  </list>
</getMemberListResponse>
```

Appendix 1 Response Code Definition

Usage: The response code that Merchant will receive are defined through the API.

Code	Message in English	Description
0	Success	Interface call is successful.
1	System Error	Internal system error at Game provider. Example: Program bug or database connection failed.
2	Invalid Request	Request is not support by Game provider.
3	Service Inaccessible	Game provider API down.
100	Request Timeout	
101	Call Limited	
104	Request Forbidden	
105	Missing Parameters	
106	Invalid Parameters	
107	Duplicated Serial NO.	
109	Related id not found	This Transfer API is usually used for a single wallet
110	Record ID Not Found	
111	Record ID Not Found	
112	API Call Limited	Exceed the limit of calling API
113	Invalid Acct ID	Acct ID incorrect format
118	Invalid Format	Parse Json Data Failed
120	IP no whitelisted	
5003	System Maintenance	
10113	Merchant Not Found	
10116	merchant suspend	
50099	Acct Exist	
50100	Acct Not Found	
50101	Acct Inactive	
50102	Acct Locked	
50103	Acct Suspend	
50104	Token Validation Failed	
50110	Insufficient Balance	
50111	Exceed Max Amount	
50112	Currency Invalid	Deposit/withdraws currency code are not consistent with Player's default currency code or merchant does not use the currency code.
50113	Amount Invalid	Deposit/withdraw Amount must be greater than 0.
50115	Date Format Invalid	Date Format incorrect format

Appendix 2 Language Code Definition

ISO Language Code	Description
en_US	English
zh_CN	Simplified Chinese
th_TH	Thai
id_ID	Indonesian
vi_VN	Vietnamese
ko_KR	Korea
ja_JP	Japan
ru_RU	Russia
tr_TR	Turkey
es_ES	Spanish
fr_FR	French
pt_PT	Portuguese

Appendix 3 Currency Code Definition (ISO)

ISO Currency Code	Description	ISO Currency Code	Description
BDT	Bangladeshi Taka	MXN	Mexican Peso
CNY	Chinese Yuan (Renminbi)	NZD	New Zealand Dollar
USD	US Dollar	PEN	Peruvian New Sol
EUR	Euro	PLN	Polish Zloty
EPV	Euro	PYG	Paraguayan Guarani
GBP	British Pound	RSD	Serbian Dinar
HKD	Hong Kong Dollar	UYU	Uruguayan Peso
SGD	Singapore Dollar	ZAR	South African Rand
KRW	South Korean Won	UAH	Ukrainian hryvnia
JPY	Japanese Yen	TND	Tunisian Dinar
THB	Thai Baht	RUB	Russian ruble
IDR	Indonesia Rupiah	TRY	Turkish Lira
ID2	Indonesia Rupiah	SEK	Swedish Krona
INR	Indian Rupee	NOK	Norwegian Krone
VND	Viet Nam Dong	AED	Dirham
VN2	Viet Nam Dong	AMD	Armenian Dram
MYR	Malaysia Ringgit	CAD	Canadian Dollar
MMK	Myanmar Kyat	CHF	Swiss franc
MMV	Myanmar Kyat	CLP	Chilean Peso
AUD	Australian Dollar	CZK	Czech Koruna
ALL	Albania	DKK	Danish Krone
BRL	Brazilian Real	ARS	Argentine Peso
BGN	Bulgarian Lev	COP	Colombian Peso
IRR	IranianRial	CO2	Colombian Peso
IR2	IranianRial	PKR	Pakistan Rupee
KZT	Kazakhstani Tenge	HUF	Hungarian Forint

Notes:

1. IRR, COP, VND, MMV, LAK, KHR and IDR currency using ratio 1:1000
2. IR2, CO2, VN2, MMK and ID2 currency using ratio 1:1
3. Please refer the latest currency in the gamelist document

Appendix 4 API Sample Codes in Java(HTTP)

Constants.java

```
public interface Constants {
    public static final Charset Charset = Charsets.UTF_8;
    public static final String  DateFormat = "yyyyMMdd'T'HHmmss";
    public static final String GZIP = "gzip";
    public static final int PageSize = 100;

    public static interface HttpHeaders {
        public static final String ACCEPT_ENCODING =
        "Accept-Encoding";
        public static final String ACCEPT_LANGUAGE =
        "Accept-Language";
        public static final String CONTENT_LENGTH =
        "Content-Length";
        public static final String CONTENT_TYPE = "Content-Type";
        public static final String CONTENT_ENCODING =
        "Content-Encoding";
        public static final String API = "API";
        public static final String DATA_TYPE = "DataType";
        public static final String DIGEST = "Digest";
    }

    public static interface DataType {
        public static final String Xml = "XML";
        public static final String Json = "JSON";
    }
}
```

Api Post.java

```
    public static byte[] post(String target, String api, String
    dataType, byte[] data) throws Exception {
     String requestHash = DigestUtils.digest( data );
     HttpURLConnection conn = null;
    OutputStream out = null;
    InputStream in = null;
     try {
         String dataLength = Integer.toString( data.length );

         //Create connection
         URL url = new URL( target );
         conn = (HttpURLConnection) url.openConnection();
         conn.setRequestMethod( "POST" );
         conn.setRequestProperty( API, api );
         conn.setRequestProperty( DATA_TYPE, dataType );
         conn.setRequestProperty( DIGEST, requestHash );
         conn.setRequestProperty( CONTENT_LENGTH, dataLength );
         conn.setRequestProperty( ACCEPT_ENCODING, GZIP );
         conn.setUseCaches( false );
         conn.setDoInput( true );
         conn.setDoOutput( true );
         conn.setReadTimeout( timeout );

         //Send request & handle Response
         out = conn.getOutputStream();
         out.write( data );
         out.flush();
```

```
            in = conn.getInputStream();
            return handleResponse( in, conn );
        } finally {
            IOUtils.closeQuietly( out );
            IOUtils.closeQuietly( in );
            if ( conn != null ) {
                conn.disconnect();
            }
        }
}

    private static byte[] handleResponse(final InputStream in,
    final HttpURLConnection conn) throws IOException {
     final int code = conn.getResponseCode();
     if ( code != 200 ) {
         throw new RuntimeException( "server return error! status "
         + code );
     }
     InputStream is = in;
     String encoding = conn.getHeaderField( CONTENT_ENCODING );
    if ( StringUtils.notEmpty( encoding ) &&
    encoding.indexOf( GZIP ) >= 0 ) {
         int contentLength = conn.getContentLength();
         is = contentLength > 0 ? new GZIPInputStream( in,
         contentLength ) : new GZIPInputStream( in );
     }
     byte[] data = IOUtils.toByteArray( is );
     String responseHash = conn.getHeaderField( DIGEST );
     if ( StringUtils.notEmpty( responseHash ) ) {
         String computedHash = DigestUtils.digest( data );
         if ( !responseHash.equals( computedHash ) ) {
             String msg = String.format( "received digest data
             invalid! response: %s", new String( data, Charset ) );
             throw new RuntimeException( msg );
         }
     }
     return data;
}
```

Appendix 5 API Sample Codes in C#(HTTP)

HttpHeader.cs

```
public class HttpHeaders {
    public const String AcceptEncoding = "Accept-Encoding";
    public const String AcceptLanguage = "Accept-Language";
    public const String ContentEncoding = "Content-
Encoding";
    public const String Api = "API";
    public const String DataType = "DataType";
    public const String Digest = "Digest";
}
```

HttpPost.cs

```
public class HttpPost
{
    public event MessageEventHandler OnMessageReceived;

   private static readonly HashAlgorithm DigestProvider =
   new MD5CryptoServiceProvider();
    private static readonly Encoding Charset =
Encoding.UTF8;

   public void Send(String target, String api, String
   dataType, String data)
    {
        Byte[] body = Charset.GetBytes(data);
        String hash = ComputeHash(body);

        HttpWebRequest request =
        (HttpWebRequest)WebRequest.Create(target);
        request.Method = "POST";
        request.ContentLength = body.Length;
        request.Headers[HttpHeaders.Api] = api;
        request.Headers[HttpHeaders.DataType] = dataType;
        request.Headers[HttpHeaders.AcceptEncoding] =
        "gzip";   //support gzip
        request.AutomaticDecompression =
        DecompressionMethods.GZip;
        request.Headers[HttpHeaders.Digest] = hash;
        using (Stream os = request.GetRequestStream())
        {
            os.Write(body, 0, body.Length);
        }

        IAsyncResult result = request.BeginGetResponse(new
        AsyncCallback(AsyncReceive), request);
    }


   private void AsyncReceive(IAsyncResult result)
    {
        HttpWebRequest request =
        (HttpWebRequest)result.AsyncState;
        HttpWebResponse response =
```

```
            (HttpWebResponse)request.EndGetResponse(result);
            String message = null;
            using (Stream stream = response.GetResponseStream())
            {
                StreamReader sr = new StreamReader(stream,
                Charset);
                message = sr.ReadToEnd().Trim();

                if (response.StatusCode != HttpStatusCode.OK)
                {
                    message += "err: " + response.StatusCode +
                    Environment.NewLine;
                }
            }

            string hash = response.GetResponseHeader("hash");
            if (!String.IsNullOrEmpty(hash)
            && !"err".Equals(message))
             {
                Byte[] body = Charset.GetBytes(message);
                String computed = ComputeHash(body);
                if (!computed.Equals(hash))
                {
                    message += String.Format("hash not equal:
                    \nreceived: {0}, computed: {1}", hash,
                    computed);
                 }
                 else
                {
                    hash += Environment.NewLine + "Hash check
                    passed";
                }
            }

            if (OnMessageReceived != null)
            {
                MessageEventArgs e = new MessageEventArgs();
                e.Data = message;
                OnMessageReceived(this, e);
            }
        }

        private String ComputeHash(Byte[] body)
        {
            Byte[] hashBytes = DigestProvider.ComputeHash(body);
                String hash =
                BitConverter.ToString(hashBytes).Replace("-",
                String.Empty);
            return hash;
        }
}
```

Appendix 6 Transfer API Example:

Type 1: Transfer Request – Place Bet


```
{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211000",
        "currency": "USD",
        "amount": 10,
        "type": 1,
        "ticketId":"0",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
}
```


Type 4: Payout (Trigger free game)


```
{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211001",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211000",
        "specialGame":
        {
          "type": "Free",
          "count": 5,
          "sequence": 0
        }
}

{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211002",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211000",
        "specialGame":
        {
          "type": "Free",
          "count": 5,
          "sequence": 1
        }
}

{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211003",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211000",
        "specialGame":
        {
          "type": "Free",
          "count": 5,
```

```
            "sequence": 2
        }
}

{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211 0004",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211 0000",
        "specialGame":
        {
          "type": "Free",
          "count": 5,
          "sequence": 3
        }
}



{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211 0005",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211 0000",
        "specialGame":
        {
          "type": "Free",
          "count": 5,
          "sequence": 4
        }
}

{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211 0006",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211 0000",
        "specialGame":
        {
          "type": "Free",
          "count": 5,
          "sequence": 5
        }
}
```

Type 4: Payout (Free game trigger bonus)


```
{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211000 1",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211000 0",
        "specialGame":
        {
          "type": "Free",
          "count": 5,
          "sequence": 0
        }
}

{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211000 2",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211000 0",
        "specialGame":
        {
          "type": "Free",
          "count": 5,
          "sequence": 1
        }
}

{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211000 3",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211000 0",
        "specialGame":
        {
          "type": "Free",
          "count": 5,
          "sequence": 2
        }
}

{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211000 4",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211000 0",
        "specialGame":
        {
          "type": "FreeBonus",
          "count": 5,
          "sequence": 3
        }
}
```

```
{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211000 5",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211000 0",
        "specialGame":
        {
          "type": "BonusFree",
          "count": 1,
          "sequence": 1
        }
}

{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211000 6",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211000 0",
        "specialGame":
        {
          "type": "Free",
          "count": 5,
          "sequence": 4
        }
}

{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211000 7",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211000 0",
        "specialGame":
        {
          "type": "Free",
          "count": 5,
          "sequence": 5
        }
}
```

Type 4: Payout (Free game trigger free game)


```
{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211008",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211000",
        "specialGame":
        {
          "type": "Free",
          "count": 5,
          "sequence": 0
        }
}

{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211009",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211000",
        "specialGame":
        {
          "type": "Free",
          "count": 5,
          "sequence": 1
        }
}

{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211010",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211000",
        "specialGame":
        {
          "type": "Free",
          "count": 10,
          "sequence": 2
        }
}

{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211011",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211000",
        "specialGame":
        {
          "type": "Free",
          "count": 10,
          "sequence": 3
        }
}
```

Sequence 4 – 9

```
{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211003 4",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "2019021417322407211000",
        "specialGame":
        {
          "type": "Free",
          "count": 10,
          "sequence": 10
        }
}
```

Type 4: Payout (Bonus game trigger free game)


```
{
        "acctId": "TESTPLAYER1",
        "transferId": "20190214173224072110008",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "20190214173224072110000",
        "specialGame":
        {
          "type": "Bonus",
          "count": 1,
          "sequence": 0
        }
}

{
        "acctId": "TESTPLAYER1",
        "transferId": "20190214173224072110009",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "20190214173224072110000",
        "specialGame":
        {
          "type": "BonusFree",
          "count": 1,
          "sequence": 1
        }
}

{
        "acctId": "TESTPLAYER1",
        "transferId": "20190214173224072110010",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "20190214173224072110000",
        "specialGame":
        {
          "type": "Free",
          "count": 10,
          "sequence": 1
        }
}

{
        "acctId": "TESTPLAYER1",
        "transferId": "20190214173224072110011",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "20190214173224072110000",
        "specialGame":
        {
          "type": "Free",
          "count": 10,
          "sequence": 2
        }
}
```

```
{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211003 4",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "201902141732240721100 00",
        "specialGame":
        {
          "type": "Free",
          "count": 20,
          "sequence": 3
        }
}

.
**Sequence 4 – 19**
.

{
        "acctId": "TESTPLAYER1",
        "transferId": "2019021417322407211001 1",
        "currency": "USD",
        "amount": 10,
        "type": 4,
        "ticketId":"234950357",
        "channel": "Web",
        "gameCode": "S-PP01",
        "merchantCode": "SPADE",
        "serialNo": "20120722231413699735",
        "referenceId": "201902141732240721100 00",
        "specialGame":
        {
          "type": "Free",
          "count": 20,
          "sequence": 20
        }
}
```

Type 4: Payout (Cascade Game)


```
{
                "acctId": "TESTPLAYER1",
                "transferId": "2019021417322407211000 8",
                "currency": "USD",
                "amount": 10,
                "type": 4,
                "ticketId":"234950357",
                "channel": "Web",
                "gameCode": "S-SW01",
                "merchantCode": "SPADE",
                "serialNo": "20120722231413699735",
                "referenceId": "2019021417322407211000 0",
                "refTicketIds":["4389578","4389579"]
}
```


Type 4: Payout (Fishing Game)


```
{
                "acctId": "TESTPLAYER1",
                "transferId": "2019021417322407211000 8",
                "currency": "USD",
                "amount": 10,
                "type": 4,
                "ticketId":"234950357",
                "channel": "Web",
                "gameCode": " F-SF01",
                "merchantCode": "SPADE",
                "serialNo": "20120722231413699735",
                "referenceId": "2019021417322407211000 0",

}
```


Type 4: Payout (Spin All)


```
{
                "acctId": "TESTPLAYER1",
                "transferId": "2019021417322407211000 8",
                "currency": "USD",
                "type": 4,
                "specialGame":
                {
                   "type":"Free",
                   "count":15,
                   "sequence":15
                },
                "amount":500,
                "ticketId":"544360560",
                "refTicketIds":
                ["544360560","544360575]
                "channel": "Web",
                "gameCode": " S-SW01",
                "merchantCode": "SPADE",
                "serialNo": "20120722231413699735",
                "referenceId": "2019021417322407211000 0",

}
```