MobiLoto Integration API

SUBSET DOCUMENTATUIN FOR Mobiloto integration API functionality PROVIDED FOR WEBSITE AND MOBILE APPLICATIONS

Table of Contents

[General API Information 4](#_Toc100309036)

[Testing Environment 4](#_Toc100309037)

[Production Environment 4](#_Toc100309038)

[Supported Access Methods 4](#_Toc100309039)

[.NET Service 4](#_Toc100309040)

[XML 4](#_Toc100309041)

[JSON 5](#_Toc100309042)

[JSONP 5](#_Toc100309043)

[Base Reply Structure 6](#_Toc100309044)

[.NET 6](#_Toc100309045)

[XML 6](#_Toc100309046)

[JSON 6](#_Toc100309047)

[Error Codes 7](#_Toc100309048)

[API Documentation 8](#_Toc100309049)

[User Login, Profile and Password Management 8](#_Toc100309050)

[LoginUser 8](#_Toc100309051)

[LogoutUser 8](#_Toc100309052)

[CheckAndRefreshToken 8](#_Toc100309053)

[User Balance and Transactions Management 9](#_Toc100309054)

[GetUserBalance 9](#_Toc100309055)

[GetUserTransactionHistory (Loto and Yawmiyeh) 9](#_Toc100309056)

[GetUserTransactionHistoryPaginated (Loto and Yawmiyeh) 10](#_Toc100309057)

[GetUserTransactionHistoryDetail (Loto and Yawmiyeh) 10](#_Toc100309058)

[GetUserLotoTransactionHistoryDetail 11](#_Toc100309059)

[GetUserYawmiyehTransactionHistoryDetail 11](#_Toc100309060)

[Draw Information 11](#_Toc100309061)

[GetNextDrawInformation 11](#_Toc100309062)

[GetInPlayDrawInformation 11](#_Toc100309063)

[GetInPlayAndNextDrawInformation 12](#_Toc100309064)

[GetAllRelevantDrawInformation 12](#_Toc100309065)

[GetLastDrawInformation 12](#_Toc100309066)

[GetDrawsInformation 13](#_Toc100309067)

[GetAllDraws 13](#_Toc100309068)

[GetDrawsPaginated 13](#_Toc100309069)

[GetBallStatisticsForCurrentYear 14](#_Toc100309070)

[Game Functionalities and Playing 14](#_Toc100309071)

[GetGridPrice 14](#_Toc100309072)

[GetFullGridPriceMatrix 14](#_Toc100309073)

[GetGridandZeedPrice 15](#_Toc100309074)

[GetDrawsWithPlayedGrids 15](#_Toc100309075)

[GetSubscriptions 15](#_Toc100309076)

[GetPlayedGridsForDraw 15](#_Toc100309077)

[CheckGridWinnings 16](#_Toc100309078)

[SubmitLotoPlayOrder (Deprecated) 16](#_Toc100309079)

[SubmitLotoPlayOrderAndClearSubcartOnly (Yawmiyeh and Loto) 17](#_Toc100309080)

# General API Information

## Testing Environment

The current testing environment currently supports only the HTTP protocol. Integrating applications can access the environment using the following link:

<http://backbonetest.lebaneseloto.com/Servicev2.asmx>

## Production Environment

The production environment will be deployed inside the LLDJ network. The final address and links are not available as of yet. The communication will be over the HTTPS protocol.

<https://backbone.lebaneseloto.com/Service.asmx>

## Supported Access Methods

In order to provide access to the most technologies possible, including integration of client-side and server-side web applications, and mobile applications for major platforms, the API provided will returns results using any of the formats listed below depending on the format of the API call.

### .NET Service

ASP.NET applications can consume the provided web service directly. The ASP.NET provides the tools necessary to generate client proxy classes that applications can use to call the available functions.

The WSDL URI of the service is:

[[http://backbonetest.lebaneseloto.com/Servicev2.asmx](https://backbonetest.lebaneseloto.com/Servicev2.asmx)?WSDL](http://newlldj.mobikick.com/Service.asmx?WSDL)

### XML

To have results of the web service functions returned in a plain XML format, applications can call the required functions using the following URL schema:

[[http://backbonetest.lebaneseloto.com/Servicev2.asmx](https://backbonetest.lebaneseloto.com/Servicev2.asmx)/[FunctionName](http://newlldj.mobikick.com/Service.asmx/%5bFunctionName)]

The integrating application can send the parameters via either POST or GET requests depending on its requirements.

Example:   
[[http://backbonetest.lebaneseloto.com/Servicev2.asmx](https://backbonetest.lebaneseloto.com/Servicev2.asmx)/GetAllDraws?Token=xxxx](http://newlldj.mobikick.com/Service.asmx/GetAllDraws?Token=xxxx)

### JSON

In order to have results formatted as JSON, applications can call the required functions using the following URL schema:

[[http://backbonetest.lebaneseloto.com/Servicev2.asmx](https://backbonetest.lebaneseloto.com/Servicev2.asmx)/[FunctionName](http://newlldj.mobikick.com/Service.asmx/%5bFunctionName)]

The request Content Type in the header should be application/json

The integrating application can only send the parameters via POST requests as part of the URL in order to get a JSONP reply. The application should format the parameters in JSON.

**N.B.: For functions that do not require any parameter, the application should send a dummy parameter in order to force the web service to return JSON. Otherwise, the service will return XML.**

Example:

[[http://backbonetest.lebaneseloto.com/Servicev2.asmx](https://backbonetest.lebaneseloto.com/Servicev2.asmx)/GetAllDraws](http://newlldj.mobikick.com/Service.asmx/GetAllDraws)

Headers: Content-type: application/json

Post Data: {“Token”:”xxxx”}

Example:

[http://backbonetest.lebaneseloto.com/Servicev2.asmx](https://backbonetest.lebaneseloto.com/Servicev2.asmx)/GetBallStatisticsForCurrentYear

Headers: Content-type: application/json

Post Data: {“Dummy”:”dummy”}

### JSONP

In order to have results via JSONP call back, applications can call the required functions using the following URL schema:

[http://backbonetest.lebaneseloto.com/Servicev2.asmx](https://backbonetest.lebaneseloto.com/Servicev2.asmx)/[FunctionName]&format=json&callback=[CallbackFunctionName]

The integrating application can only send the parameters via POST requests as part of the URL in order to get a JSONP reply. The application should format the parameters in JSON.

Example: [[http://backbonetest.lebaneseloto.com/Servicev2.asmx](https://backbonetest.lebaneseloto.com/Servicev2.asmx)/GetAllDraws?format=json&callback=test](http://newlldj.mobikick.com/Service.asmx/GetAllDraws?format=json&callback=test) with Post data: {“Token”:”xxxx”}

## Base Reply Structure

### .NET

The class ***BaseReply*** is the base class for all the objects returned by the web service.

***BaseReply*** contains an ***ErrorInfo*** class. This class holds both an *errorcode* integer indicating the error code number and an *errormessage* string returning an Error Message. The application can show the error message to the user as is or it can process it depending on its needs. *errorcode* will be 0 upon success.

Depending on the function called, the calling application will need to cast the BaseReply object it received to the appropriate object. Refer to function documentation for more information.

### XML

The element <BaseReply> is the root element for all the replies returned by the web service. It will always hold an <errorinfo> element that will hold both <errorcode> holding an error code number and <errormsg> holding an error message. The application can show the error message to the user as is or it can process it depending on its needs. Errorcode will be 0 upon success.

Depending on the function called the application will return other elements within the root element. Refer to function documentation for more information.

### JSON

“d” will be the name of the root Json object returned by the web service. It will always contain an “errorinfo” object. “errorcode” and “errormsg” contain the error code returned by the web service along with an error message. The application can show the error to the user as is or it can process it depending on its needs. The “errorcode” 0 indicates success.

Depending on the function called, the application will return other elements within the root element. Refer to function documentation for more information.

## Error Codes

|  |  |
| --- | --- |
| **Code** | **Description** |
| 0 | Success |
| 1 | Invalid Username or Password |
| 2 | Token Expired |
| 3 | Not Enough Credit |
| 4 | Not Allowed |
| 5 | Invalid Voucher Code |
| 6 | Invalid Parameter |
| 7 | Username Exists |
| 8 | User is Blocked |
| 9 | Invalid Grid |
| 10 | Not Implemented Yet |
| 11 | Required Field Not Supplied |
| 12 | User Not Activated |
| 13 | Email Already in Use |
| 14 | User Exceeded Limit |
| 15 | Username Cannot Be Emiail |
| 16 | Cannot change email because identical usename |
| 17 | Code 104 Already Used |
| 18 | Code 104 Not Available |
| 10000 | Generic Error |

Any Web Service function can return the Generic Error Code. This usually means that there is a temporary issue with the Database. If the same call repeatedly returns this error, then we might need to trace that call.

# API Documentation

## User Login, Profile and Password Management

### LoginUser

Description: This function allows application to sign in the user with his username or email and password. In return, the function will return the first and last name of the user, his balance and a token used in all subsequent calls to the web service to identify the current user. This token will be valid for 30 minutes from the last call to the web service that contains this token.

N.B: For Social Networks Login, the calling application will generate the username using an identifier for the social network and the social network ID (Refer to RegisterUser function for more information). The password should be a valid token from the social network that our server can verify. Other functions will use the token returned by this function.

Parameters: Username (Can be the username or the email interchangeably), Password

Returned Information: firstname, lastname, balance, token

Possible Error Codes: 1 (Invalid Username or password), 8 (User is Blocked)

### LogoutUser

Description: This function allows application to sign out the user. It will invalidate the token and all subsequent calls to function requiring Token authentication will receive the Invalid Token error code.

Parameters: Token

Returned Information: Only Error Code and Success.

Possible Error Codes: 2 (Token Expired)

### CheckAndRefreshToken

Description: This function will check whether a token has expired or not. If it hasn’t, it will be refreshed.

Parameters: Token

Returned Information: Only Error Code or Success

Possible Error Codes: 2 (Token Expired)

## User Balance and Transactions Management

### GetUserBalance

Description: This function will return the account balance for the logged in user

Parameters: Token

Returned Information: token, balance

Possible Error Codes: 2 (Token Expired)

### GetUserTransactionHistory (Loto and Yawmiyeh)

Description: This function will return all the logged in user transaction history between 2 dates, filtered by transaction type.

Parameters: Token, fromDate, toDate, transactionType (0 for all transaction types)

Returned Information: An Array of History Entries (historyId, description, date, amount, isCredit, typeId, isCommitted, comment, mailSent, mttCode, tempInfo, idIni, zeed, fee).

Possible Error Codes: 2 (Token Expired)

Type Codes:

|  |  |
| --- | --- |
| 1 | Loto Played |
| 2 | Recharge |
| 3 | Winnings |
| 4 | Refund |
| 5 | Cash Deposit |
| 6 | Check Deposit |
| 7 | Credit Card Deposit |
| 8 | Wire Transfer Deposit |
| 9 | Check Withdrawal |
| 10 | Wire Transfer Withdrawal |
| 11 | Nothing (Profile) |
| 12 | Cash Retrieval |
| 13 | Vouchers And Coupons |
| 16 | Delayed Credit Card Transaction |

Special Filters:

|  |  |
| --- | --- |
| 1 | All Game Operations |
| 1000 | All Debit or Credit Operations |
| 1001 | All Money Deposit or Withdrawal Operations |
| 1002 | All Profile Operations Such As Change Password |

### GetUserTransactionHistoryPaginated (Loto and Yawmiyeh)

Description: This function will return all the logged in user transaction history between 2 dates, filtered by transaction type. Provide Starting Transaction Count (offset) and Number of transactions to return (limit)

Parameters: Token, fromDate, toDate, transactionType (check above function for list of possible transaction Types), offset, limit

Returned Information: An Array of History Entries (historyId, description, date, amount, isCredit, typeId, isCommitted, comment, mailSent, mttCode, tempInfo, idIni, zeed, fee).

Possible Error Codes: 2 (Token Expired)

### GetUserTransactionHistoryDetail (Loto and Yawmiyeh)

Description: This function will return detailed information about a specific transaction involving playing Loto or Yawmiyeh Grids for the logged in user. (typeId=1 or 100) . For Loto, It can returns this info as played by the user (Bouquet Id=0) or as a breakdown of actual grids played on the loto system. (Bouquet Id = -1)

It can also return the specific grids played in any bouquet in any order. (By Specifying actual Bouquet Id retrieved from the listing with Bouquet Id=0)

Parameters: Token, historyId, bouquetId (return grids for specific bouquet if > 0 )

Returned Information: The List of Played Grids for the specified transaction History Id or a specific Bouquet Id. isFavorite field is always null in this call  
 gridId: use as BouquetId for Bouquet Grids.  
 gridBalls: Holds the balls played or “Bouquet of xxx”  
 drawNumber: draw number for Played Grids

Additional Fields when Loto played and bouquet Id = -1 or greater than 0

drawDate: draw date  
 gridPrice: Price of that grid at time of Playing  
 zeedNumber: the Zeed Number associated with the grid or -1 if no Zeed.  
  
Specific Yawmiyeh Notes:   
 bouquetId parameter is discarded.   
 zeedNumber is always empty.  
 drawDate: will contain the draw date if the grid is played for the current or any older Yawmiyeh draw. Upcoming draws will have an empty draw date.

Possible Error Codes: 2 (Token Expired)

### GetUserLotoTransactionHistoryDetail

Description: This function will return detailed information about a specific transaction involving playing Loto Grids for the logged in user. (typeId=1) . It can returns this info as played by the user (Bouquet Id=0) or as a breakdown of actual grids played on the loto system. (Bouquet Id = -1). Check above function for more information

### GetUserYawmiyehTransactionHistoryDetail

Description: This function will return detailed information about a specific transaction involving playing Yawmiyeh Grids for the logged in user. (typeId=100). Check above 2 functions for more information.

## Draw Information

### GetNextDrawInformation

Description: This function gets Jackpot and Draw information for upcoming draw.

Parameters: Token (Can Be Empty)

Returned Information: A BasicDrawInfo element with the following fields: drawnumber, drawdate, lotojackpotUSD, zeedjackpotUSD, lotojackpotLBP, zeedjackpotLBP, orderTransactionFee

Possible Error Codes: Generic Error Only

### GetInPlayDrawInformation

Description: This function gets the actual draw in which played grids will participate. It can be different then the next draw after playing for a certain draw has closed and until that draw is completed. The calling application should discard the jackpot information from this call and instead, should use the jackpot information from the above function.

Parameters: Token (Can Be Empty)

Returned Information: A BasicDrawInfo element with the following fields: drawnumber, drawdate, lotojackpotUSD, zeedjackpotUSD, lotojackpotLBP, zeedjackpotLBP, orderTransactionFee

Possible Error Codes: Generic Error Only

### GetInPlayAndNextDrawInformation

Description: This function gets both the actual draw in which played grids will participate and the last draw that was closed with Jackpot information combining the above two functions

Parameters: Token (Can Be Empty)

Returned Information: Two BasicDrawInfo elements with the following fields: drawnumber, drawdate, lotojackpotUSD, zeedjackpotUSD, lotojackpotLBP, zeedjackpotLBP, orderTransactionFee

Possible Error Codes: Generic Error Only

### GetAllRelevantDrawInformation

Description: This function gets both the actual draw in which played grids will participate and the last draw that was closed with Jackpot information. It also returns the last draw information.

Parameters: Token (Can Be Empty)

Returned Information: Two BasicDrawInfo elements with the following fields: drawnumber, drawdate, lotojackpotUSD, zeedjackpotUSD, lotojackpotLBP, zeedjackpotLBP, orderTransactionFee   
And a BasicDrawInfo element with the following fields: drawnumber, drawdate, B1, B2, B3, B4, B5, B6, B7, prize1, winners1, prize2, winners2, prize3, winners3, prize4, winners4, prize5, winners5, haszeed, zeednumber, zeednumber2,zeednumber3, zeednumber4, zeedjackpot, zeednextjackpot, zeedwinners1, zeedwinners2, zeedwinners3, zeedwinners4, zeedprize1, zeedprize2, zeedprize3, zeedprize4, videolink

Possible Error Codes: Generic Error Only

### GetLastDrawInformation

Description: This function gets last draw winning information including Zeed Winners, and Prizes. The calling application will use this function to display the latest results pages.

Parameters: Token (Can Be Empty)

Returned Information: A BasicDrawInfo element with the following fields: drawnumber, drawdate, B1, B2, B3, B4, B5, B6, B7, prize1, winners1, prize2, winners2, prize3, winners3, prize4, winners4, prize5, winners5, haszeed, zeednumber, zeednumber2,zeednumber3, zeednumber4, zeedjackpot, zeednextjackpot, zeedwinners1, zeedwinners2, zeedwinners3, zeedwinners4, zeedprize1, zeedprize2, zeedprize3, zeedprize4, videolink

Possible Error Codes: Generic Error Only

### GetDrawsInformation

Description: This function gets a list of all draws from specific Draw Number to another specific Draw Number. This list is extensive and contains all the information for the listed draws. This function only returns a maximum of 10 draws, otherwise returns a not allowed error. If both from and to parameters are 0, then the function will return the last 10 closed draws.

Parameters: Token (Can Be Empty), from (can be 0 along with the to parameter), to

Returned Information: An Array of BasicDrawInfo elements with the following fields: drawnumber, drawdate, B1, B2, B3, B4, B5, B6, B7, prize1, winners1, prize2, winners2, prize3, winners3, prize4, winners4, prize5, winners5, haszeed, zeednumber, zeednumber2,zeednumber3, zeednumber4, zeedjackpot, zeednextjackpot, zeedwinners1, zeedprize1, zeedprize2, zeedprize3, zeedprize4, videolink

Possible Error Codes: 4 (Not Allowed)

### GetAllDraws

Description: This function gets a list of all draw numbers and dates only. To get full information, the calling application should call the above function.

Parameters: Token (Can Be Empty)

Returned Information: An array of BasicDrawInfo elements with the following fields: drawnumber, drawdate

Possible Error Codes: Generic Error Only

### GetDrawsPaginated

Description: This function gets a list of all draw numbers and dates only. To get full information, the calling application should call the above function. Provide Starting Draw Count (offset) and Number of draws to return (limit)

Parameters: Token (Can Be Empty), Offset, Limit

Returned Information: An array of BasicDrawInfo elements with the following fields: drawnumber, drawdate

Possible Error Codes: Generic Error Only

### GetBallStatisticsForCurrentYear

Description: This function will return the ball count for each of the 42 lotto balls during the current calendar year.

Parameters: No Parameters Required

Returned Information: Id, Index, DrawCount, Year, B1Count to B42Count

Possible Error Codes: Generic Error Only

## Game Functionalities and Playing

### GetGridPrice

Description: This function calculates and returns the price of the grid parameter. This Grid should be a sequence of ball numbers separated by space, sorted by ascending order. Minimum 6 Balls, Maximum 12 Balls. Jokers Ball Values = 43, 44. Furthermore, the grid can be in the format of Bxxx where xxx is the actual number of grids to be played randomly by the system. Ex: B50 to play 50 random grids. Note that for invalid grids, this function will not return an error but a grid price of 0.

Parameters: Token, Grid

Returned Information: stringvalue1 (Grid Price), stringvalue2 (Zeed Price)

Possible Error Codes: Only Generic

### GetFullGridPriceMatrix

Description: This function calculates and returns the price of all possible grid of 6,7,8,9 and 10 balls with 0, 1 or 2 Jockers. It also returns the Unit price of a 6 ball grid and that of the Zeed Addition.

Parameters: Token, Grid

Returned Information: unitprice (Grid Price), zeedprice (Zeed Price), an array of PriceMatrixForBallNum (ballNum, price0J, price1J, price2J)

Possible Error Codes: Only Generic

### GetGridandZeedPrice

Description: This function returns the grid unit price (The price of a base 6 number grid) and the Zeed unit price.

Parameters: No Parameters Required

Returned Information: stringvalue1 (Grid Price), stringvalue2 (Zeed Price)

Possible Error Codes: Only Generic

### GetDrawsWithPlayedGrids

Description: This function will return all the draws where the logged in user has already played grids for.

Parameters: Token

Returned Information: An array of BasicDrawInfo elements with the following fields: drawnumber, drawdate

Possible Error Codes: 2 (Token Expired)

### GetSubscriptions

Description: This function will return all the subscriptions for the logged in user. A subscription is any order where the user plays grids for more than one draw.

Parameters: Token

Returned Information: An array of SubscriptionInfo elements with the following fields: orderid, startdrawnumber, enddrawnumber,remainingdraws , drawcount, gridcountperdraw, startdate, enddate

Possible Error Codes: 2 (Token Expired)

### GetPlayedGridsForDraw

Description: This function will return all the grids that the logged in user played at a specific draw or at the last order for the last closed draw he actually participated in.

Parameters: Token, drawNumber (Can be 0 to retrieve played grids for last order played for last closed draw by user)

Returned Information: An Array of Grid Elements containing gridId, gridBalls, and gridPrice. isFavorite parameter of grid object is always null in this call (Refer to GetGridprice function above for info on how grids are returned)

Possible Error Codes: 2 (Token Expired)

### CheckGridWinnings

Description: Use this function to check specific grids winnings from the grid bar code. The Bar Code can be the 12-digit code, or the 22-digit code with verification, with or without dashes.

**N.B: This web service might not implement this feature on first launch. If this is the case, it will return an error message to the user indicating that this feature is not yet available. Currently, for testing purposes, it will work with grid number 000000000000 to 000000000003. Otherwise, it returns an error. The calling application should display the message to the user as is. Each of the grids enabled returns a different scenario (Combination of Lotto or Zeed Winnings)**

Parameters: GridNumber

Returned Information: stringvalue1 (Grid Winnings), stringvalue2 (Zeed Winnings), drawnumber, drawdate, issmsgrid

Possible Error Codes: 10 (Not Implemented Yet)

### SubmitLotoPlayOrder (Deprecated)

Description: *This function is deprecated. Use SubmitLotoPlayOrderAndClearSubcartOnly function for playing a single order directly from the Play page with Sequence Parameter set to -1. Use the SubmitPlayOrder to submit orders directly from cart page.*   
  
This function saves a lotto Play Order for Immediate Processing for the logged in user. Returns either an error message or the user balance in case the order is successful. It can also save all the played grids (except the bouquets of random grids) to the user favorite grids. Please refer to parameters for more information. Note that the User Cart will be cleared upon submitting any order.

Parameters: Token: Token for Logged In User  
 drawNumber: Draw Number for which to Play the grids  
 numDraws: Number of Draws for which to play the grids (at least 1)  
 withZeed: 1 to Add Zeed to the order.  
 saveToFavorite: 1 To Save All Played Grids To Favorites  
 GridsSelected: A List of Grids Separated by | (The grids should be in the format specified in the GetGridPrice function above)

Returned Information: token, balance, insertId (will contain selected Zeed Number if order is played with Zeed)

Possible Error Codes: 2 (Token Expired), 4 (Not Allowed if draw has closed), 9 (Invalid Grid), 6 (Invalid Parameter), 14 (User Exceeded Limit)

### SubmitLotoPlayOrderAndClearSubcartOnly (Yawmiyeh and Loto)

Description: Same as the above function. Note that the system will only clear the specified sub-Cart upon submitting an order with this function.

Parameters: Token: Token for Logged In User  
 drawNumber: Draw Number for which to Play the grids  
 numDraws: Number of Draws for which to play the grids (at least 1)  
 withZeed: 1 to Add Zeed to the order.  
 saveToFavorite: 1 To Save All Played Grids To Favorites  
 GridsSelected: A List of Grids Separated by | (The grids should be in the format specified in the GetGridPrice function above)  
 Sequence: Cart Sequence or subcart identifier to be cleared from cart.

Returned Information: token, balance, insertId (will contain selected Zeed Number if order is played with Zeed)

Possible Error Codes: 2 (Token Expired), 4 (Not Allowed if draw has closed), 9 (Invalid Grid), 6 (Invalid Parameter), 14 (User Exceeded Limit)