

|  |
| --- |
|  |
| Mobile Authentication Corporation |
| Test Bank Services API Version 1.5 |

|  |
| --- |
|  |

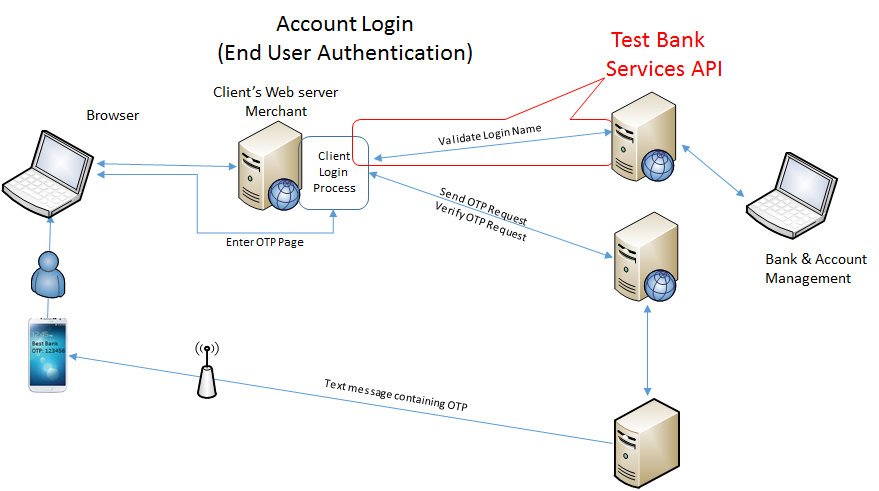
# Introduction

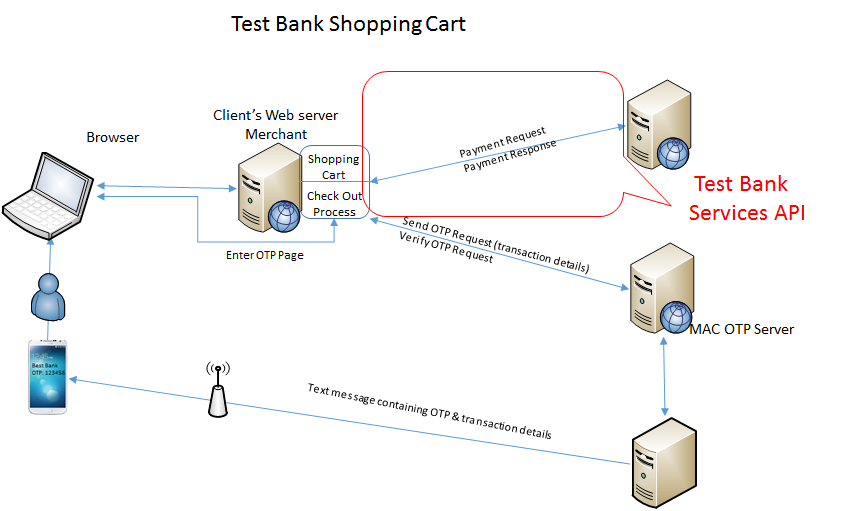
This document defines the API for the MAC Test Bank Web Services. The Test Bank was created provide a “bank like” simulated backend that could be used by the MAC demos The front end web based test and demo applications will use the MAC Test bank services to simulate account access and funds management and movement to provide a verifiable OTP process.

Overview

Contained in this document are brief descriptions of the HTTP/HTTPS methods for sending requests to the Test Bank Services, the responses, the message formats, and the client requirements. The document also contains JavaScript/JQuery examples of the functions which create and issue the calls, as well as, process the responses.

## Usage diagrams





## Test Bank Features

The Test bank supports accounts that are numbered using the Standard Account and Card numbering scheme as to facilitate the interfacing with existing shopping carts, see the Account and Card Numbering section for more details. The Test bank supports credit card, debit card, prepaid account and funds movement that exists all within the test bank.

### Bank Account Profile

A Bank account is accessed via a Primary Account Number (PAN) or the Account Holder’s Name (First Last).

There are 7 pre allocated accounts and subaccounts each having a specific use:

* Deposit (PAN Balance)
* Pre-paid account, balance can’t go negative,
* Credit card account, balance can go negative, open usage(any merchant)
* Debit card account, balance can’t go negative, open usage(any merchant)
* Private credit card account, balance can go negative, restricted usage,
* Private debit card account, balance can’t go negative, restricted usage,
* Client restricted credit card account, closed usage.

## Service API Requests

Services support the HTTP/HTTPS Post method where the data contains the request details. Depending on the implementation, the data is either converted to hexadecimal before sending as data to the request, this avoids special character conflicts.

***Note:*** *MAC Test Server URL: corp.mobileauthcorp.com/macservices/Test/MacTestBank.asmx/WsMacTestBank*

***Note:*** *For the MAC Test Bank Calls get the Default client id from the web.config:*

*public static string DefaultClientId = ConfigurationManager.AppSettings["DefaultClientId"];*

## Request Format Details

The parameters for a request are assembled in an ASCII string as a key value pare with each key/value separated by the pipe character “|” shown as “dk.**ItemSep”** in the following code examples and the key value is separated by the colon character “:” shown as dk.**KVSep** in the following code examples. If the value “could” contains a colon character, the value must be converted to a hexadecimal string before it is added to the request parameters.

**Note:** The MAC Test Bank uses email addresses for login Ids.

### Request / Response Details

* **Get MAC Test Bank Status** – Request the bank information, number of accounts, number of accounts in assigned, list of PANs, list of account holders, list of account and sub-account names and a list of login names.

var rply = SendRequestToMacTestBankServer(DefaultClientId, dk.**Request** + dk.**KVSep** + "GetBankStatus");

* + **Example of xml formatted response**

***Note:*** *the following depends on the current bank configuration.*

<?xml version="1.0" encoding="utf-8"?>

<macResponse totalProcessTime="196ms">

<calledMethod>WsMacTestBank()</calledMethod>

<Reply>Success</Reply>

<TotalAccounts>17</TotalAccounts>

<AssignedAccounts>17</AssignedAccounts>

<PANList>6601000000000208|6601000000000307|6601000000000406</PANList>

<AccountHoldersList>John Tester1:User|Bob Tester2:User|Sam Tester3:User</AccountHoldersList>

<AccountNamesList>PAN|Prepaid Account|Credit Card|Debit Card|Group Credit Card|Group Debit Card|Client Card</AccountNamesList>

<LoginNamesList>Tester1@elbsoftware.com|Tester2@elbsoftware.com|Tester3@elbsoftware.com </LoginNamesList>

</macResponse>

* **Validate Login Name** - Request login sent to Test Bank service with login id, response will contain user's UserId.

var rply = SendRequestToMacTestBankServer(DefaultClientId,

dk.**Request** + dk.**KVSep** + "ValidateLoginName" +

dk.**ItemSep** + "LoginName" + dk.**KVSep** + LoginName;

* + **Example of xml formatted response**

<?xml version="1.0" encoding="utf-8"?>

<macResponse totalProcessTime="2ms">

<calledMethod>WsMacTestBank()</calledMethod>

<Reply>Success</Reply>

<Details>6601000000001008, John Tester1</Details>

</macResponse>

* **Deposit** – Deposit funds into an account or sub-Account.

Note: Deposit uses “CreditAccount” request verb.

* + **Example using account holder name**

var rply = SendRequestToMacTestBankServer(DefaultClientId,

dk.**Request** + dk.**KVSep** + "CreditAccount" +

dk.**ItemSep** + "AccountHolder" + dk.**KVSep** + mAccountHolderName +

dk.**ItemSep** + "AccountName" + dk.**KVSep** + mAccountName +

dk.**ItemSep** + "Amount" + dk.**KVSep** + amount;

* **Withdrawal** – Withdrawal funds from an account.

Note: Deposit uses “DebitAccount” request verb.

* + **Example using account holder name**

var rply = SendRequestToMacTestBankServer(DefaultClientId,

dk.**Request** + dk.**KVSep** + "DebitAccount" +

dk.**ItemSep** + "AccountHolder" + dk.**KVSep** + mAccountHolderName +

dk.**ItemSep** + "AccountName" + dk.**KVSep** + mAccountName +

dk.**ItemSep** + "Amount" + dk.**KVSep** + amount;

* + **Example of xml formatted response**

<?xml version="1.0" encoding="utf-8"?>

<macResponse totalProcessTime="4ms">

<Reply>Success</Reply>

<Details>PAN account credited for $1,500.00, new balance: $1,500.00</Details>

</macResponse>

* **Get Account Status** – Request Account status sent to Test Bank service with PAN, response contains account and subaccount details.
  + **Example using PAN**

var rply = SendRequestToMacTestBankServer(DefaultClientId,

dk.**Request** + dk.**KVSep** + "GetAccountDetails" +

dk.**ItemSep** + "AccountNo" + dk.**KVSep** + PAN;

* + **Example using Account holder name**

var rply = SendRequestToMacTestBankServer(DefaultClientId,

dk.**Request** + dk.**KVSep** + "GetAccountDetails" +

dk.**ItemSep** + "AccountHolder" + dk.**KVSep** + AccountHolderName;

* + **Example of xml formatted response**

<?xml version="1.0" encoding="utf-8"?>

<macResponse totalProcessTime="3ms">

<calledMethod>WsMacTestBank()</calledMethod>

<Reply> Success</Reply>

<Details>

<Account>

<PAN>6601000000000109</PAN>

<Balance>$1,400.00</Balance>

<Created>5/28/2014 3:20:34 PM</Created>

<Updated>1/1/0001 12:00:00 AM</Updated>

<AccountHolder>Joe Doe</AccountHolder>

<SubAccounts>

<SubAccount>

<Name>Prepaid Account</Name>

<Number>6601010000000108</Number>

<Enabled>True</Enabled>

<LastAccessed>5/28/2014 3:20:27 PM</LastAccessed>

<Balance>$4,499.75</Balance>

<Limit>$10,000.00</Limit>

</SubAccount>

<SubAccount>

<Name>Credit Card</Name>

<Number>6601020000000107</Number>

<Enabled>True</Enabled>

<LastAccessed>5/28/2014 3:20:27 PM</LastAccessed>

<Balance>$0.00</Balance>

<Limit>$1,000.00</Limit>

</SubAccount>

<SubAccount>

<Name>Debit Card</Name>

<Number>6601030000000106</Number>

<Enabled>True</Enabled>

<LastAccessed>5/28/2014 3:20:27 PM</LastAccessed>

<Balance>$0.00</Balance>

<Limit>$10,000.00</Limit>

</SubAccount>

<SubAccount>

<Name>Group Credit Card</Name>

<Number>6601040000000105</Number>

<Enabled>True</Enabled>

<LastAccessed>5/28/2014 3:20:27 PM</LastAccessed>

<Balance>$0.00</Balance>

<Limit>$1,000.00</Limit>

</SubAccount>

<SubAccount>

<Name>Group Debit Card</Name>

<Number>6601050000000104</Number>

<Enabled>True</Enabled>

<LastAccessed>5/28/2014 3:20:27 PM</LastAccessed>

<Balance>$0.00</Balance>

<Limit>$10,000.00</Limit>

</SubAccount>

<SubAccount>

<Name>Client Card</Name>

<Number>6601060000000103</Number>

<Enabled>True</Enabled>

<LastAccessed>5/28/2014 3:20:27 PM</LastAccessed>

<Balance>$0.00</Balance>

<Limit>$10,000.00</Limit>

</SubAccount>

</SubAccounts>

<ActivityLog>

<LogEntry>5/28/2014 3:20:34 PM Assigned as User to Terry Davis</LogEntry>

<LogEntry>5/28/2014 7:33:17 PM Funds Movement $100.00 from 6601000000000109.PAN to 6601000000000109.Prepaid Account</LogEntry>

<LogEntry>5/28/2014 7:56:36 PM Funds Movement $100.00 from 6601000000000109.PAN to 6601000000000208.PAN</LogEntry>

<LogEntry>6/1/2014 4:01:04 PM Funds Movement $100.25 from 6601000000000109.Prepaid Account to 6601000000000307.PAN</LogEntry>

</ActivityLog>

</Account>

</Details>

</macResponse>

* **Transfer funds** - Transfer funds request sent to Test Bank service with from account/sub-account number, to account/sub-account number and amount, response successful or reason for failure. Note: uses the “MoveFunds” request verb.

var rply = SendRequestToMacTestBankServer(DefaultClientId,

dk.**Request** + dk.**KVSep** + "MoveFunds" +

dk.**ItemSep** + "FromAccountHolder" + dk.**KVSep** + FromAccountHolderName +

dk.**ItemSep** + "FromAccountName" + dk.**KVSep** + FromAccountName +

dk.**ItemSep** + "ToAccountHolder" + dk.**KVSep** + ToAccountHolderName +

dk.**ItemSep** + "ToAccountName" + dk.**KVSep** + ToAccountName +

dk.**ItemSep** + "Amount" + dk.**KVSep** + AmountToMove);

* + **Example of XML formatted response**

<?xml version="1.0" encoding="utf-8"?>

<macResponse totalProcessTime="1ms">

<calledMethod>WsMacTestBank()</calledMethod>

<Reply>Success</Reply>

<Details>Funds Movement $100.25 from 6601000000001008.PAN to 6601000000001008.Prepaid Account</Details>

</macResponse>

* **Preauthorization of payment** - Preauth request sent to Test Bank service with account/sub-account number and amount, response Funds avaliable or Insuffient funds.

var rply = SendRequestToMacTestBankServer(DefaultClientId,

dk.**Request** + dk.**KVSep** + "Preauth" +

dk.**ItemSep** + "AccountNo" + dk.**KVSep** + AccountNumber +

dk.**ItemSep** + "Amount" + dk.**KVSep** + AmountToMove);

* + **Example of XML formatted response**

<?xml version="1.0" encoding="utf-8"?>

<macResponse totalProcessTime="1ms">

<calledMethod>WsMacTestBank()</calledMethod>

<Reply>Success</Reply>

<Details>Funds available</Details>

</macResponse>

* **Get Bills** – Part of the Bill Payment calls, Get Bills returns a list of bills for an account holder.
  + **Example using account holder name**

var rply = SendRequestToMacTestBankServer(DefaultClientId,

dk.**Request** + dk.**KVSep** + "GetBills" +

dk.**ItemSep** + "AccountHolder" + dk.**KVSep** + AccountHolderName);

* + **Example using account number**

var rply = SendRequestToMacTestBankServer(DefaultClientId,

dk.**Request** + dk.**KVSep** + "GetBills" +

dk.**ItemSep** + "AccountNo" + dk.**KVSep** + AccountNumber);

* + **Example of XML formatted response**

<?xml version="1.0" encoding="utf-8"?>

<macResponse totalProcessTime="134ms">

<calledMethod>WsMacTestBank()</calledMethod>

<Reply>Success</Reply>

<Account>

<PAN>6601000000000109</PAN>

<Bills>

<Bill>

<InvoiceNumber>10</InvoiceNumber>

<Status>Due</Status>

<BusinessType>Utility</BusinessType>

<Name>The Electric Company</Name>

<BillingDate>7/11/2014 12:08:06 AM</BillingDate

<AmountDue>$1,212.00</AmountDue>

<DueDate>7/26/2014 12:08:06 AM</DueDate>

</Bill>

</Bills>

</Account>

</macResponse>

* **Bill payment** – Bill payment request sent to Test Bank service with merchant name, payment method as account of subaccount number or name and amount. Note: uses the “PayBill” request verb.
  + **Example using account holder name**

var rply = SendRequestToMacTestBankServer(DefaultClientId,

dk.**Request** + dk.**KVSep** + "PayBill" +

dk.**ItemSep** + "InvoiceNumber" + dk.**KVSep** + InvoiceNumberToPay +

dk.**ItemSep** + "AccountHolder" + dk.**KVSep** + AccountHolderName +

dk.**ItemSep** + "AccountName" + dk.**KVSep** + FromAccountName +

dk.**ItemSep** + "Amount" + dk.**KVSep** + AmountToPay);

* + **Example of XML formatted response**

<?xml version="1.0" encoding="utf-8"?>

<macResponse totalProcessTime="7ms">

<calledMethod>WsMacTestBank()</calledMethod>

<Reply>

<Details>

Success: Bill from Target paid from Credit Card[6601020000000107] payment in the amount of $30.00 remaining balance $195.00

</Details>

</Reply>

</macResponse>

* **Purchase** – Purchase request is used when the Account holder is buying something from a merchant.
  + **Example request**

var rply = SendRequestToMacTestBankServer(DefaultClientId,

dk.**Request** + dk.**KVSep** + "Purchase" +

dk.**ItemSep** + "MerchantName" + dk.**KVSep** + MerchantName +

dk.**ItemSep** + "AccountHolder" + dk.**KVSep** + AccountHolderName +

dk.**ItemSep** + "CardNumber" + dk.**KVSep** + AccountNumber + // null if PAN

dk.**ItemSep** + "Amount" + dk.**KVSep** + pAmount +

dk.**ItemSep** + "TrxDetails" + dk.**KVSep** + StringToHex(pDetails));

* + **Example of XML formatted response**

<?xml version="1.0" encoding="utf-8"?>

<macResponse totalProcessTime="300ms">

<calledMethod>WsMacTestBank()</calledMethod>

<Reply>Success</Reply>

<Details>

Purchase Kohl's, Card:6601020000000107, Amount:$100.00, Details:NA

</Details>

</macResponse>

* **Get Account Holder’s Account Names and Numbers** – The request is used to get a list of Accounts and Sub-Accounts and their BIN numbers.
  + **Example request**

var rply = SendRequestToMacTestBankServer(DefaultClientId,

dk.**Request** + dk.**KVSep** + "GetAccountNamesNumbers" +

dk.**ItemSep** + "AccountHolder" + dk.**KVSep** + AccountHolderName);

* + **Example of XML formatted response**

<?xml version="1.0" encoding="utf-8"?>

<macResponse totalProcessTime="210ms">

<calledMethod>WsMacTestBank()</calledMethod>

<Reply>Success</Reply>

<Details>

PAN:6601000000000109|Prepaid Account:6601010000000108|Credit Card:6601020000000107|Debit Card:6601030000000106|Group Credit Card:6601040000000105|Group Debit Card:6601050000000104|Client Card:6601060000000103

</Details>

</macResponse>

* **Get Account transaction log** – Get account transaction log request sent to Test Bank service with PAN.

Note: uses the “GetAccountLog” request verb.

* + **Example using PAN**

var rply = SendRequestToMacTestBankServer(DefaultClientId,

dk.**Request** + dk.**KVSep** + "GetAccountLog" +

dk.**ItemSep** + "AccountNo" + dk.**KVSep** + PAN);

* + **Example using account holder name**

var rply = SendRequestToMacTestBankServer(DefaultClientId,

dk.**Request** + dk.**KVSep** + "GetAccountLog" +

dk.**ItemSep** + "AccountHolder" + dk.**KVSep** + AccountHolder);

* + **Example of XML formatted response**

<?xml version="1.0" encoding="utf-8"?>

<macResponse totalProcessTime="3ms">

<calledMethod>WsMacTestBank()</calledMethod>

<Reply>

<Details>

Success:

<Account>

<PAN>6601000000000109</PAN>

<ActivityLog>

<LogEntry>

5/28/2014 3:20:34 PM Assigned as User to Joe Doe

</LogEntry>

<LogEntry>

5/28/2014 7:33:17 PM Funds Movement $100.00 from 6601000000000109.PAN to 6601000000000109.Prepaid Account

</LogEntry>

<LogEntry>

5/28/2014 7:56:36 PM Funds Movement $100.00 from 6601000000000109.PAN to 6601000000000208.PAN

</LogEntry>

<LogEntry>

6/1/2014 4:01:04 PM Funds Movement $100.25 from 6601000000000109.Prepaid Account to 6601000000000307.PAN

</LogEntry>

<LogEntry>

6/1/2014 4:38:01 PM Funds Movement $100.25 from 6601000000000109.Prepaid Account to 6601000000000109.Debit Card

</LogEntry>

<LogEntry>

6/1/2014 4:38:01 PM Funds Movement $100.25 from 6601000000000109.Prepaid Account to 6601000000000109.Debit Card

</LogEntry>

</ActivityLog>

</Account>

</Details>

</Reply>

</macResponse>

## Account and Card Numbering

An ISO/IEC 7812 card number is most commonly 16 digits in length,[1] and consists of a six-digit Issuer Identification Number (IIN) (previously called the "Bank Identification Number" (BIN)) the first digit of which is the Major Industry Identifier (MII), variable length (up to 12 digits) individual account identifier and a single check digit calculated using the Luhn algorithm.[2]

For MAC Test bank a sixteen digit number: 6601 0TAA AAAA AAUC

Six-digit Issuer Identification Number (IIN)

6601 0T, where T is the type account or card

0 = primary account number

1 = prepaid account

2 = Credit card

3 = Debit card

4 = Private label debit card

Eight-digit individual account identifier (AA AAAA AA)

00 0000 00 - 99 9999 99

Single digit cardholder number (U)

Single check digit (C)

# Change History

Version 1.1 – Original document

Version 1.2 – After initial testing

Version 1.3 – Added Preauth request to service

Version 1.4 – Correct the document, ValidateLoginName request used LoginName not LoginId

Version 1.5 – Added Get Account Holder’s Account Names and Numbers **“**GetAccountNamesNumbers”and Purchase **“**Purchase” requests to service