



MOBBIS LLC , 1/21/2009

.NET API 3.0

Functional specification



1. Introduction.....	4
1.1. Document identifier.....	4
1.2. Scope.....	4
1.3. Definition of terms.....	4
2. References.....	4
3. Overview.....	4
4. .NET API structure.....	4
4.1. SMSMessage.....	4
4.2. Account class.....	5
4.2.1. Public properties.....	5
4.2.2. Methods.....	6
4.3. Message class.....	7
4.3.1. Methods.....	7
5. Appendix.....	10
5.1. Message statuses.....	10

1. Introduction

1.1. Document identifier

This is the specification of the .NET API 3.0 implemented by MOBBIS LLC.

1.2. Scope

This document contains the description of the methods provided by the .NET API, which allow to manipulate with the accounts and process message(s) through MOBIPACE SMS Gateway.

It also contains the definition of the terms and acronyms used in the text and references to the useful links.

1.3. Definition of terms

HTTPS - <http://en.wikipedia.org/wiki/HTTPS>

SMS - http://en.wikipedia.org/wiki/Short_message_service

MOBIPACE - <http://www.mobipace.com>

ASCII - <http://en.wikipedia.org/wiki/ASCII>

UTC - <http://en.wikipedia.org/wiki/UTC>

UNICODE - <http://en.wikipedia.org/wiki/Unicode>

Web Service - http://www.webopedia.com/TERM/W/Web_services.html

WSDL - http://en.wikipedia.org/wiki/Web_Services_Description_Language

2. References

[1]. <http://www.mobipace.com>

[2]. <http://www.mobbis.mobi>

[3]. <http://www.mobbis.am>

[4]. http://en.wikipedia.org/wiki/List_of_Unicode_characters

[5]. <http://en.wikipedia.org/wiki/ASCII>

[6]. HTTP_API_Specification

[7]. XML_API_Specification

3. Overview

.NET API DLL is a wrapper for the HTTP API (see [6]) and XML API (see [7]). It contains the methods which allow to manage the accounts and process messages through MOBIPACE Gateway.

4. .NET API structure

The API is implemented under the namespace "MOBBIS.NetApi".The classes included in the library are described below.

4.1. SMSMessage

The class represents the message which should be processed.

The properties are given below.

/// <summary>

```

/// Specifies the cell phone number of the message recipient
/// </summary>
public string Recipient;

/// </summary>
/// Represents the body of the message (both ASCII and UNICODE are supported)
/// </summary>
public string Body;

/// </summary>
/// Specifies whether the message should be considered as Wap Push or not
/// </summary>
public bool IsWapPush;

/// </summary>
/// Represents the name which should be used as sender name for the message
/// </summary>
public string SenderName;

```

4.2. Account class

This class contains methods to register/activate/delete MOBIPACE accounts. It also contains fields which are being used for performing MOBIPACE Gateway and network Proxy authentication.

4.2.1. Public properties

```

/// </summary>
/// Specifies whether the request should bypass Proxy server
/// </summary>
public bool ProxyEnabled;

/// </summary>
/// Specifies the username of the MOBIPACE Gateway user account
/// </summary>
public string Username;

/// </summary>
/// Specifies the password of the MOBIPACE Gateway user account
/// </summary>
public string Password;

/// </summary>
/// Specifies the proxy server address
/// </summary>
public string ProxyAddress;

/// </summary>
/// Specifies the port of the Proxy server
/// </summary>
public int ProxyPort;

```

Username - the login name of the account. Is used for MOBIPACE Gateway authentication when processing messages and for registration of a new account by it.

Password - the password of the account.

ProxyEnabled - Specifies whether the HTTP requests should bypass Proxy authentication.

ProxyAddress - Specifies the address of the Proxy server.

ProxyPort - Specifies the port of the Proxy server.

4.2.2. Methods

All methods require the properties of an object of type Account to be properly specified. The encountered errors are being reported via exceptions, which are derived from the **MOBBIS.NetAPI.BaseException**, which provides a method "GetInfoMessage" returning the string representing the exception. The all methods throw exception **MOBBIS.NetAPI.WebRequestError** when some error occurs during accessing the web resources, e.g. Incorrect Proxy server configuration etc.

◆ RegisterUserAccount

The method creates a new inactive account by the specified username/password. The activation code is being sent to the cell phone number specified as username. If the type of the currency is not specified, the "EUR" will be considered as the default currency type. The default language will be set to ENG, which can be changed in the future from the MOBIPACE's web interface (www.mobipace.com).

This request works for the individual persons only, which username is the cell phone numbers. Refer to www.mobipace.com differences between individual and juridical types of the users.

The information about the errors occurred during the registration process are being reported by the exceptions. The signature of the method is:

```
public void RegisterUserAccount(string Email, string Question, string Answer, string Currency)
```

where:

Email specifies the e-mail address which can be used for password recovery in further,

Question specifies the control question which can be used for password recovery in further,

Answer specifies the answer to the control question,

Currency specifies the desired currency type (can be one of "EUR", "USD", "RUR", "AMD",),).

The possible exceptions which can be thrown in case of errors are:

MOBBIS.NetAPI.UserNameIsAlreadyUsed - is thrown in case when the specified username by which user is trying to register is already used,

MOBBIS.NetAPI.IncorrectMailFormat - is thrown when the format of the specified e-mail is incorrect,

MOBBIS.NetAPI.IncorrectCurrency - is thrown when the specified currency is not supported,

MOBBIS.NetAPI.UserNameIsMissing - is thrown when the username is not initialized,

MOBBIS.NetAPI.PasswordIsMissing - is thrown when the password is not initialized,

MOBBIS.NetAPI.EmailsAlreadyUsed - is thrown when user account with the specified email address is already registered,

MOBBIS.NetAPI.IncorrectParameterValue - is thrown when one of the specified parameter values is incorrect. This exception is a generic exception and mostly specifies some issue uncaught by the system.

◆ ActivateUserAccount

Activates an already registered inactive user account by the specified activation code. This method is supposed to be called when the previous method has already been called and an inactive user account is already registered.

The already registered user account is valid for the web based project MOBIPACE (www.mobipace.com) also, provides an interface to send/schedule and view the statuses of the messages.

The signature of the function is:

```
public void ActivateUserAccount(int Code)
```

where the **Code** is the activation code.

The possible exceptions are:

MOBBIS.NetAPI.AuthenticationFailed – is thrown when trying to activate an incorrect user account,

MOBBIS.NetAPI.IncorrectActivationCode – is thrown when the specified activation code is incorrect,

MOBBIS.NetAPI.UserNameIsMissing – is thrown when the username is not initialized,

MOBBIS.NetAPI.PasswordIsMissing – is thrown when the password is not initialized,

MOBBIS.NetAPI.IncorrectParameterValue – is thrown when one of the specified parameter values is incorrect. This exception is a generic exception and mostly specifies some issue uncaught by the system.

◆ DeleteUserAccount

This method deletes an already registered user account. The signature of the method is:

```
public void DeleteUserAccount()
```

The possible exceptions are:

MOBBIS.NetAPI.AuthenticationFailed – is thrown when trying to delete an incorrect user account, i.e. user account by the specified username and password is not registered,

MOBBIS.NetAPI.UserNameIsMissing – is thrown when the username is not initialized,

MOBBIS.NetAPI.PasswordIsMissing – is thrown when the password is not initialized.

◆ QueryBalance

Returns the current balance of the user in the type of the currency set for the specified user.

The signature of the method is:

```
public double QueryBalance()
```

The possible errors are being informed via exceptions described below.

MOBBIS.NetAPI.AuthenticationFailed – is thrown when the specified user account is not valid,

MOBBIS.NetAPI.UserNameIsMissing – is thrown when the username is not initialized,

MOBBIS.NetAPI.PasswordIsMissing – is thrown when the password is not initialized,

4.3. Message class

This class contains methods and members to process messages. The class contains the property “Account” to set/get an object of the Account specifying the user account by which the processing of the messages will be done.

4.3.1. Methods

All methods require the account property to be properly specified. The encountered errors are being reported via exceptions, which are derived from the **MOBBIS.NetAPI.BaseException**, which provides a method “GetInfoMessage” returning the string representing the exception.

MOBBIS.NetAPI.WebRequestError when some error occurs during accessing the web resources, e.g. Incorrect Proxy server configuration etc.

◆ SendMessage

Send the message through the MOBBIS gateway to the cell phone specified as recipient. It is possible to specify the type of the message – Wap Push or SMS.

If the size of one message exceeds 160 symbols (the maximum length of the SMS message), the message will be automatically split into corresponding count of messages and then sent to the recipient sequentially.

It is possible to send both UNICODE (see [\[4\]](#)) and ASCII (see [\[5\]](#)) type of messages.

Note, that the message will be considered as UNICODE message if it contains even one UNICODE symbol. In this case the count of the messages can be increased.

Note, for registered individual persons the sender field of the message will be set to username (cell phone number).

The signature of the method is:

```
public string SendMessage(MOBBIS.NetApi.Message m)
```

where **m** is the object of type MOBBIS.NetApi.Message representing a single message (see [4.1](#) for details).

The method returns message reference (ID) generated by the system to be used in further for requesting the status of the message.

The possible errors are being informed via exceptions described below.

MOBBIS.NetAPI.AuthenticationFailed - is thrown when the specified user account is not valid,

MOBBIS.NetAPI.OutOfBalance - is thrown when the balance of the user is not enough to send the message,

MOBBIS.NetAPI.IncorrectRecipient - is thrown when the recipient number is not valid cell phone number,

MOBBIS.NetAPI.UserNameIsMissing - is thrown when the username is not initialized,

MOBBIS.NetAPI.PasswordIsMissing - is thrown when the password is not initialized,

MOBBIS.NetAPI.RecipientIsMissing - is thrown when the recipient is not specified,

MOBBIS.NetAPI.IncorrectParameterValue - is thrown when one of the specified parameter values is incorrect. This exception is a generic exception and mostly specifies some issue uncaught by the system.

◆ ScheduleMessage

This method schedules the message to be sent on specified date/time to the specified recipient. Note, that the date/time should be specified in UTC international format. The function returns the message reference of the scheduled message to handle the message in the future.

If the size of one message exceeds 160 symbols (the maximum length of the SMS message), the message will be automatically split into corresponding count of messages and then sent to the recipient sequentially.

It is possible to send both UNICODE (see [\[4\]](#)) and ASCII (see [\[5\]](#)) type of messages.

Note, that the message will be considered as UNICODE message if it contains even one UNICODE symbol. In this case the count of the messages can be increased.

Note, for registered individual persons the sender field of the message will be set to username (cell phone number).

Signature of the method is:

```
public string ScheduleMessage(MOBBIS.NetApi.Message m, DateTime d)
```

where

- **m** is the object of type MOBBIS.NetApi.Message representing a single message (see [4.1](#) for details),
- **d** is the scheduling date/time in UTC

The methods returns message reference(ID) generated by the system to be used in further for requesting the status of the message.

The possible errors are being informed via exceptions described below.

MOBBIS.NetAPI.AuthenticationFailed - is thrown when the specified user account is not valid,

MOBBIS.NetAPI.OutOfBalance - is thrown when balance of the user is not enough to send the message,

MOBBIS.NetAPI.IncorrectRecipient - is thrown when the recipient number is not valid cell phone number,

MOBBIS.NetAPI.IncorrectParameterValue – is thrown when one of the specified parameter values is incorrect. This exception is a generic exception and mostly specifies some issue uncaught by the system,

MOBBIS.NetAPI.IncorrectDateTimeFormat – is thrown when the format of the date/time is not correct,

MOBBIS.NetAPI.OutOfDateTime – is thrown when the specified date/time is out of date,

MOBBIS.NetAPI.UserNameIsMissing – is thrown when the username is not initialized,

MOBBIS.NetAPI.PasswordIsMissing – is thrown when the password is not initialized,

MOBBIS.NetAPI.MessageContentIsEmpty – is thrown when the message body is empty,

MOBBIS.NetAPI.RecipientIsMissing – is thrown when the recipient is not specified.

◆ ScheduleMessages

This method schedules a train of messages to be sent on specified date/time.

There are two methods which signatures are:

```
public string[] ScheduleMessages(MOBBIS.NetApi.Message[] m, DateTime[] d)
```

where

- **m** is an array of the objects of type MOBBIS.NetApi.Message (see [4.1](#) for details),
- **d** is an array containing scheduling date/time in UTC of the corresponding message,

```
public string[] ScheduleMessages(MOBBIS.NetApi.Message[] m, DateTime d)
```

where

- **m** is an array of the objects of type MOBBIS.NetApi.Message (see [4.1](#) for details)
- **d** is the scheduling date/time in UTC of the whole train of messages

The method returns array with message references (IDs) of the messages generated by the system to be used in further for requesting the status of the message.

The possible errors are being informed via exceptions described below.

MOBBIS.NetAPI.AuthenticationFailed – is thrown when the specified user account is not valid,

MOBBIS.NetAPI.OutOfBalance – is thrown when balance of the user is not enough to send the message,

MOBBIS.NetAPI.IncorrectRecipient – is thrown when the recipient number is not valid cell phone number,

MOBBIS.NetAPI.IncorrectParameterValue – is thrown when one of the specified parameter values is incorrect. This exception is a generic exception and mostly specifies some issue uncaught by the system,

MOBBIS.NetAPI.IncorrectDateTimeFormat – is thrown when the format of the date/time is not correct,

MOBBIS.NetAPI.OutOfDateTime – is thrown when the specified date/time is out of date,

MOBBIS.NetAPI.UserNameIsMissing – is thrown when the username is not initialized,

MOBBIS.NetAPI.PasswordIsMissing – is thrown when the password is not initialized,

MOBBIS.NetAPI.MessageContentIsEmpty – is thrown when the message body is empty,

MOBBIS.NetAPI.RecipientIsMissing – is thrown when the recipient is not specified.

◆ QueryMessage

Returns the status of the message by the specified message reference.

The signature of the method is:

```
public MOBBIS.NetApi.Definitions.MessageStatus QueryMessage(string MessageRef)
```

where the **MessageRef** is the reference (ID) of the message which status is being requested.

The method returns an objects of type **MOBBIS.NetApi.MessageStatus** (see [5.1](#)) defining the status of the message.

The possible errors are being informed via exceptions described below.

MOBBIS.NetApi.AuthenticationFailed - is thrown when the specified user account is not valid,

MOBBIS.NetApi.UserNameIsMissing - is thrown when the username is not initialized,

MOBBIS.NetApi.PasswordIsMissing - is thrown when the password is not initialized,

MOBBIS.NetApi.IncorrectMessageRef - is thrown when the specified message reference is not valid.

◆ QueryMessages

Returns the statuses of a train of message by the specified message references.

The signature of the method is:

```
public MOBBIS.NetApi.MessageStatus[] QueryMessages(string[] MessageRefs)
```

where the **MessageRefs** is an array of references (ID) of the message which statuses are being requested.

The method returns an array with the objects of type **MOBBIS.NetApi.MessageStatus** (see [5.1](#)) defining the status of the appropriate message.

The possible errors are being informed via exceptions described below.

MOBBIS.NetApi.AuthenticationFailed - is thrown when the specified user account is not valid,

MOBBIS.NetApi.UserNameIsMissing - is thrown when the username is not initialized,

MOBBIS.NetApi.PasswordIsMissing - is thrown when the password is not initialized,

MOBBIS.NetApi.IncorrectMessageRef - is thrown when the specified message reference is not valid.

5. Appendix

5.1. Message stasuses

Status	Value	Description
Pending	0	The status of the SMS message is unknown, SMS is in queue.
Sent	1	The SMS message is sent and delivered.
DeliveredToNetwork	2	The SMS message is delivered to operator.
TimeOut	3	The connection is interrupted, operator may be overloaded.
OutOfBalance	4	Out of balance, SMS is not delivered.
Scheduled	5	The SMS message will be sent at the scheduled time.
DeliveredToGateway	6	The SMS message is delivered to gateway but status unknown.
Rejected	7	Rejected, may be wrong destination or sender id.
NotSent	8	Wrong destination or internal error.
Unknown	9	Incorrect message ID

