SMS Gateway Interface

SMS Interface 1
Send SMS using http post Interface 1
Simple XML example 1
Advanced XML Example 2
Send Multiple Requests 3
Response XML 4
HTTP Post code sample 5
C# example 5
PHP example 7
VB Example 8
Send SMS Using Web Service Interface 10
C# example 11
Send SMS Using Parameters 12
Example for a UTF8 request: 12
Message character length 13
Delivery Notification 13
Option 1(PUSH) 13
Option 2 (PULL) 14
Receiving SMS (optional) 16
Option 1(PUSH) 16
Option 2 (PULL) 17
SMS Tools 19
Delete Future Messages 19
Remaining SMS Quota 20
Phone Number block check 21
Blocking Phone Numbers 22
Contacts & Groups Management 23
Create or Update contact 23
Example simple 23
Response 23
Example Advanced 24
Check Status Token 26
Response 26

SMS Interface

Send SMS using http post Interface

Client should perform Http Post request to this URL.
The encoding should be UTF-8.

http://uapi.inforu.co.il/SendMessageXml.ashx?InforuXML={xml}

Secure interface

For a secure interface use:

https://uapi.inforu.co.il/SendMessageXml.ashx?InforuXML={xml}

Simple XML example

```
<Inforu>
  <User>
    <Username>MyUsername</Username>
    <Password>MyPassword</Password>
  </User>
  <Content Type="sms">
    <Message>This is a test SMS Message</Message>
  </Content>
  <Recipients>
    <PhoneNumber>0501111111;0502222222</PhoneNumber>
  </Recipients>
  <Settings>
    <Sender>0501111111</Sender>
  </Settings>
</Inforu>
```

Explanation of XML structure:

- **UserName** – Username of the account that was supplied by InforUMobile
- **Password** – Password of the account
- **Message** – SMS message that needs to be sent
- **PhoneNumber** – The recipients' phone list. Can be multiple recipients separated by a semicolon ";"
- **Sender** – The ID that will be displayed in the recipient's phone as the sender.

Advanced XML Example

```
<Inforu>
  <User>
    <Username>MyUserName</Username>
    <Password>MyPassword</Password>
  </User>
  <Content Type="sms">
    <Message>This is a test SMS Message</Message>
  </Content>
  <Recipients>
    <PhoneNumber>0501111111;0502222222</PhoneNumber>
    <GroupNumber>5</GroupNumber>
  </Recipients>
  <Settings>
    <Sender>MyCompany</Sender>
    <CustomerMessageID>112233</CustomerMessageID>
    <CustomerParameter>AffId4</CustomerParameter>
    <MessageInterval>0</MessageInterval>
    <TimeToSend>12/05/2013 12:23</TimeToSend>
    <DelayInSeconds>60</DelayInSeconds>
    <DeliveryNotificationUrl>http://mysite.co.il/Notif.aspx</DeliveryNotificationUrl>
    <MaxSegments>0</MaxSegments>
    <Priority>0</Priority>
  </Settings>
</Inforu>
```

Explanation of XML structure (All the following parameters are **optional**):

- **Sender** – The "Sender" field is restricted to a maximum of 11 Latin characters (Consecutive characters, no spaces) or 14 digits (no characters). You can also write an asterisk (*) at the beginning of the identifier.
- **GroupNumber** – Use this parameter in order to send messages to a group according to the group number defined in the web site. To send a message to multiple groups please use a semicolon ";".
- **CustomerMessageID** – Message ID on the client application. When confirmation on delivery is sent back to the client, the message ID is also sent for synchronization.
- **CustomerParameter** – Parameter set by the client that can be seen in the reports, later. Can be used to mark each message by activity segmentation, for example.
- **Priority** – In order to prioritize your messages, you can use the priority tag, when 0 means normal priority, -1 high priority (used only for a single message), and then 1,2 and so on. The higher the value, the lower the priority.
- **MessageInterval** - Sending messages with number of seconds interval between them. The value 0 means non-interval.
- **DeliveryNotificationUrl** – Needed in order to send a confirmation on delivery to the client. The system will perform HTTP post to this URL with notification information.
- **MaxSegments** – When sending long messages, this parameter allows the client to set the maximum number of segments per message. Value 0 means unlimited segments.
- **TimeToSend** – Date and time on which the messages will be sent. Please use the following format: dd/mm/yyyy hh:mm. If left blank, the message will be sent immediately.
- **DelayInSeconds** – Number of seconds of delay from receiving the request in the system and up to sending the message (use this parameter as long as the TimeToSend parameter is not in use).

Send Multiple Requests

In order to send requests using different wordings, use InforuRoot – It is recommended to send up to 100 packages per request.

```
<InforuRoot>
    <Inforu>
        <User>
            <Username>MyUsername</Username>
            <Password>MyPassword</Password>
        </User>
        <Content Type="sms">
            <Message>This is the first SMS message</Message>
        </Content>
        <Recipients>
            <PhoneNumber>0501111111;0502222222</PhoneNumber>
        </Recipients>
        <Settings>
            <Sender>0501111111</Sender>
        </Settings>
    </Inforu>
    <Inforu>
        <User>
            <Username>MyUsername</Username>
            <Password>MyPassword</Password>
        </User>
        <Content Type="sms">
            <Message>This is the second SMS message</Message>
        </Content>
        <Recipients>
            <PhoneNumber>054-6669999</PhoneNumber>
        </Recipients>
        <Settings>
            <Sender>Company</Sender>
        </Settings>
    </Inforu>
</InforuRoot>
```

Response XML

Explanation of Response, XML structure:

```
<Result>
    <Status>1</Status >
      <Description>Message accepted successfully</ Description>
    <NumberOfRecipients>1</ NumberOfRecipients>
</Result>
```

(In case of using Inforuroot, the above response will repeat itself as many times as there are requests which you have inputted into the InforuRoot)

1. **Status:**
   - 1 = OK
   - -1 = Failed
   - -2 = Bad user name or password
   - -6 = RecipientsDataNotExists
   - -9 = MessageTextNotExists
   - -11 = IllegalXML
   - -13 = UserQuotaExceeded
   - -14 = ProjectQuotaExceeded
   - -15 = CustomerQuotaExceeded
   - -16 = WrongDateTime
   - -17 = WrongNumberParameter
   - -18 = No valid recipients
   - -20 = InvalidSenderNumber
   - -21 = InvalidSenderName,
   - -22 = UserBlocked
   - -26 = UserAuthenticationError
   - -28 = NetworkTypeNotSupported
   - -29 = NotAllNetworkTypesSupported
   - -90 = InvalidSenderIdentification
2. **Description –** Status interpretation.
3. **NumberOfRecipients -** Phone numbers that the message has been sent to.

HTTP Post code sample

C# example

```
//set password, user name, message text, semder name and number
string userName = "UserName";
string password = "UserPassword";
string messageText = System.Security.SecurityElement.Escape("Message text");
string sender = "MySender";



//set phone numbers
string phonesList = "0503333333;0503333334;0503333335;0503333336;0503333337";

//set additional parameters
string timeToSend = "21/12/2017 15:30";

// create XML
StringBuilder sbXml = new StringBuilder();
sbXml.Append("<Inforu>");
sbXml.Append("<User>");
sbXml.Append("<Username>" + userName + "</Username>");
sbXml.Append("<Password>" + password + "</Password>");
sbXml.Append("</User>");
sbXml.Append("<Content Type=\"sms\">");
sbXml.Append("<Message>" + messageText + "</Message>");
sbXml.Append("</Content>");
sbXml.Append("<Recipients>");
sbXml.Append("<PhoneNumber>" + phonesList + "</PhoneNumber>");
sbXml.Append("</Recipients>");
sbXml.Append("<Settings>");
sbXml.Append("<Sender>" + sender+ "</Sender>");
sbXml.Append("<MessageInterval>" + messageInterval + "</MessageInterval>");
sbXml.Append("<TimeToSend>" + timeToSend + "</TimeToSend>");
sbXml.Append("</Settings>");
sbXml.Append("</Inforu >");
string strXML = HttpUtility.UrlEncode(sbXml.ToString(), System.Text.Encoding.UTF8);
string result = PostDataToURL("http://uapi.inforu.co.il/SendMessageXml.ashx", "InforuXML=" + strXML);


static string PostDataToURL(string szUrl, string szData)
{
   //Setup the web request
   string szResult = string.Empty;
   WebRequest Request = WebRequest.Create(szUrl);
   Request.Timeout = 30000;
   Request.Method = "POST";
   Request.ContentType = "application/x-www-form-urlencoded";

    //Set the POST data in a buffer
   byte[] PostBuffer;
   try
```

```
{
    // replacing " " with "+" according to Http post RPC
    szData = szData.Replace(" ", "+");

    //Specify the length of the buffer
    PostBuffer = Encoding.UTF8.GetBytes(szData);
    Request.ContentLength = PostBuffer.Length;

    //Open up a request stream
    Stream RequestStream = Request.GetRequestStream();

    //Write the POST data
    RequestStream.Write(PostBuffer, 0, PostBuffer.Length);

     //Close the stream
    RequestStream.Close();
   //Create the Response object
   WebResponse Response;
   Response = Request.GetResponse();

   //Create the reader for the response
  StreamReader sr = new StreamReader(Response.GetResponseStream(),     Encoding.UTF8);

   //Read the response
   szResult = sr.ReadToEnd();

    //Close the reader, and response
   sr.Close();
   Response.Close();

   return szResult;
}
catch (Exception e)
{
     return szResult;
 }
}
```

Using Dot Net Programming:

- Message text must be escaped with System.Security.SecurityElement.Escape.
- XML must be encoded by HttpUtility.UrlEncode(xml, .Text.Encoding.UTF8)

PHP example

```
<?php

function SendSMS($message_text,$recepients) {
	$sms_user =		"USERNAME";		// User Name (Provided by Inforu)
	$sms_password =	"PASSWORD";		// Password (Provided by Inforu)
	$sms_sender =		"MyCompany";		//

	$message_text = preg_replace( "/\r|\n/", "", $message_text); // remove line breaks

	$xml = '';
	$xml .= '<Inforu>'.PHP_EOL;
	$xml .= ' <User>'.PHP_EOL;
	$xml .= '  <Username>'.htmlspecialchars($sms_user).'</Username>'.PHP_EOL;
	$xml .= '  <Password>'.htmlspecialchars($sms_password).'</Password>'.PHP_EOL;
	$xml .= ' </User>'.PHP_EOL;
	$xml .= ' <Content Type="sms">'.PHP_EOL;
	$xml .= '  <Message>'.htmlspecialchars($message_text).'</Message>'.PHP_EOL;
	$xml .= ' </Content>'.PHP_EOL;
	$xml .= ' <Recipients>'.PHP_EOL;
	$xml .= '  <PhoneNumber>'.htmlspecialchars($recepients).'</PhoneNumber>'.PHP_EOL;
	$xml .= ' </Recipients>'.PHP_EOL;
	$xml .= ' <Settings>'.PHP_EOL;
	$xml .= '   <Sender>'.htmlspecialchars($sms_sender).'</Sender>'.PHP_EOL;
	$xml .= ' </Settings>'.PHP_EOL;
	$xml .= '</Inforu>';

	$ch = curl_init();
	curl_setopt($ch,CURLOPT_URL,'http://uapi.inforu.co.il/SendMessageXml.ashx?InforuXML='.
	urlencode($xml));
	curl_setopt($ch,CURLOPT_RETURNTRANSFER,1);
	$response = curl_exec($ch);
	curl_close($ch);
	return $response;

	}

// example
SendSMS(
	'Hello',
	'0501111111;0502222222'
	);

?>
```

VB Example

```
//Important: need to add reference to System.Web.dll

'set password, user name, message text, semder name and number
Dim userName As String = "UserName"
Dim password As String = "UserPassword"
Dim messageText As String = System.Security.SecurityElement.Escape("Message text")
Dim sender As String = "MySender"
Dim messageInterval As Integer = 0

'set phone numbers
Dim phonesList As String = "0503333333;0503333334;0503333335;0503333336;0503333337"

'set additional parameters
Dim timeToSend As String = "21/12/2017 15:30"

' create XML
Dim sbXml As New Text.StringBuilder()
sbXml.Append("<Inforu>")
sbXml.Append("<User>")
sbXml.Append("<Username>" & userName & "</Username>")
sbXml.Append("<Password>" & password & "</Password>")
sbXml.Append("</User>")
sbXml.Append("<Content Type=""sms""">")
sbXml.Append("<Message>" & messageText & "</Message>")
sbXml.Append("</Content>")
sbXml.Append("<Recipients>")
sbXml.Append("<PhoneNumber>" & phonesList & "</PhoneNumber>")
sbXml.Append("</Recipients>")
sbXml.Append("<Settings>")
sbXml.Append("<Sender>" & sender & "</Sender>")
sbXml.Append("<MessageInterval>" & messageInterval & "</MessageInterval>")
sbXml.Append("<TimeToSend>" & timeToSend & "</TimeToSend>")
sbXml.Append("</Settings>")
sbXml.Append("</Inforu >")

Dim strXML As String = System.Web.HttpUtility.UrlEncode(sbXml.ToString(),
System.Text.Encoding.UTF8)
Dim result As String = PostDataToURL("http://uapi.inforu.co.il/SendMessageXml.ashx", "InforuXML=" &
strXML)

Private Function PostDataToURL(ByVal szUrl As String, ByVal szData As String) As String
        'Setup the web request
        Dim szResult As String = String.Empty
        Dim Request As Net.HttpWebRequest
        Request = CType(System.Net.WebRequest.Create(szUrl), System.Net.HttpWebRequest)
        Request.Timeout = 30000
        Request.Method = "POST"
        Request.ContentType = "application/x-www-form-urlencoded"

        'Set the POST data in a buffer
        Dim PostBuffer As Byte()
```

```
        Try
                ' replacing " " with "+" according to Http post RPC
                szData = szData.Replace(" ", "+")


                'Specify the length of the buffer
                PostBuffer = System.Text.Encoding.UTF8.GetBytes(szData)
                Request.ContentLength = PostBuffer.Length

                'Open up a request stream
                Dim RequestStream As IO.Stream = Request.GetRequestStream()

                'Write the POST data
                RequestStream.Write(PostBuffer, 0, PostBuffer.Length)

                'Close the stream
                RequestStream.Close()
                'Create the Response object
                Dim Response As Net.HttpWebResponse
                Response = CType(Request.GetResponse(), System.Net.HttpWebResponse)

                'Create the reader for the response
                Dim sr As New IO.StreamReader(Response.GetResponseStream(),
System.Text.Encoding.UTF8)

                'Read the response
                szResult = sr.ReadToEnd()

                'Close the reader, and response
                sr.Close()
                Response.Close()

                Return (szResult)
        Catch e As Exception
                Return (szResult)
        End Try
End Function
```

Send SMS Using Web Service Interface

Description

Web service interface is mainly used for .Net applications although Java developers can use it too. Microsoft .Net framework has a built-in support for calling web services, so for anyone who uses .Net it is highly recommended to use this interface. This interface allows the user to control and customize every aspect of the InforuMobile SMS engine.

Format

Microsoft .Net users should add a reference to this URL:

http://uapi.inforu.co.il/SendMessage.asmx

Java / Delphi / Magic users should use this URL:

http://uapi.inforu.co.il/SendMessage.asmx?wsdl

The web interface implements 2 functions:

- SendSms – Sending SMS with minimum parameters
- SendSmsDetailed – Sending SMS with additional parameters

Function Parameters Explanation

There are 2 functions for sending SMS messages.
Here is an explanation of the parameters from the function **SendSmsDetailed**:

- **UserName** - Username of the account that was supplied by InforUMobile
- **Password** - Password of the account.
- **Message** – SMS message that needs to be sent.
- **MessagePelephone** – Not in use (Use double quotation marks).
- **MessageCellcom** – Not in use (Use double quotation marks).
- **MessageOrange** – Not in use (Use double quotation marks).
- **MessageMirs** - Not in use (Use double quotation marks).
- **Recipients** – The recipients' phone list. Can be multiple recipients separated by a ";"
- **CustomerParameter** - Parameter set by the client that can be seen in the reports, later. Can be used to mark each message by activity segmentation, for example.
- **CustomerMessageID** - Message ID on the client application. When confirmation on delivery is sent back to the client, the message ID is also sent for synchronization.
- **MessageInterval** - Sending messages with number of seconds interval between them. The value 0 means non-interval.
- **TimeToSend** – Date and time on which the messages will be sent. Please use the following format: dd/mm/yyyy hh:mm. If left blank, the message will be sent immediately.
- **SenderName** – The name that will be displayed on the recipient's device, is restricted to a maximum of 11 Latin characters (Consecutive characters without spaces).
- **SenderNumber** – Not in use. (Use 0000)
- **MaxSegments** – When sending long messages, this parameter allows the client to set the maximum number of segments per message. Value 0 means unlimited segments.

C# example

```
static void Main(string[] args)
{
WS_SendSMS.SendMessageSoapClient SendSMS = new WS_SendSMS.SendMessageSoapClient();
String Result = SendSMS.SendSmsDetailed("MyUserName","MyPassword","Test using
WS","","", "","","0529999999","","",0,new
DateTime(1900,01,01,00,00,00),"MyCompany","0000", 0);
Console.WriteLine(Result);
Console.ReadLine();
}
```

Successful response example

```
<Result>
    <Status>1</Status>
    <Description>Message accepted successfully</Description>
    <NumberOfRecipients>1</NumberOfRecipients>
</Result>
```

Unsuccessful response example

```
<Result>
    <Status>-2</Status>
    <Description>Error: Bad user name or password</Description>
    <NumberOfRecipients>0</NumberOfRecipients>
</Result>
```

Response status explanations

1. **Status:**
   - 1 = OK
   - -1 = Failed
   - -2 = Bad user name or password
   - -6 = RecipientsDataNotExists
   - -9 = MessageTextNotExists
   - -11 = IllegalXML
   - -13 = UserQuotaExceeded
   - -14 = ProjectQuotaExceeded
   - -15 = CustomerQuotaExceeded
   - -16 = WrongDateTime
   - -17 = WrongNumberParameter
   - -18 = No valid recipients
   - -20 = InvalidSenderNumber
   - -21 = InvalidSenderName,
   - -22 = UserBlocked
   - -26 = UserAuthenticationError
   - -28 = NetworkTypeNotSupported
   - -90 = InvalidSenderIdentification
2. **Description –** Status interpretation.
3. **NumberOfRecipients -** Phone numbers that the message has been sent to.

Send SMS Using Parameters

Description

In order to send messages through the use of chaining parameters, this is a brief explanation of implementing the code properly.

Format

For UTF8 encoding, use the following URL:

http://uapi.inforu.co.il/inforufrontend/WebInterface/SendMessageByNumber.aspx

For WIN1255 encoding, use the following URL:

http://uapi.inforu.co.il/InforuFrontEnd/EngineInterface/SendMessageByNumber.aspx

Parameters required for the reference:

- UserName
- Password
- SenderCellNumber
- CellNumber
- MessageString

Example for a UTF8 request:

http://uapi.inforu.co.il/inforufrontend/WebInterface/SendMessageByNumber.aspx?UserName=myusername&Password=100301&SenderCellNumber=Mysender&CellNumber=052000000;0542152743&MessageString=Hello

Parameters: Explanation of Functions

- **UserName** - Username of the account that was supplied by InforUMobile
- **Password** - Password of the account
- **SenderCellNumber** – The "SenderCellNumber" field is restricted to a maximum of 11 Latin characters (Consecutive characters with no spaces) or up to 14 digits (and no characters). You can also write an asterisk (*) at the beginning of the identifier.
- **CellNumber** – Recipient's phone list. Multiple recipients may be added when separated by a semicolon ";"
- **MessageString** – SMS message that needs to be sent.

Response

The response of your requests be a number
1 = OK
Other statuses, see page 4.

Message character length

Updated information can be found here

Delivery Notification

There are two options for Notifications:

1. Push - We appeal to your URL real time.
2. Pull - You turn to us for pulling your notifications

Option 1(PUSH)

The client can receive confirmation on delivery on each message sent. When sending the message the client should use the CustomerMessageID and the DeliveryNotificationUrl parameters. The system performs an **HTTP Post** request to this URL regarding any message.

The parameters for the request are:

- **PhoneNumber** – The number of the recipient.
- **Network** – The network of the recipient.
- **Status** – The status of the message.
  - (2) – Delivered.
  - (-2) – Not delivered.
  - (-4) – Blocked by InforuMobile.
- **StatusDescription** – If not delivered, contains the reason.
- **ProjectId** – The project ID Below describes the user who sent the message.
- **CustomerMessageId** – The client message ID.
- **CustomerParam** – The client parameter.
- **SenderNumber** – The ID of the sender who sent the message.
- **SegmentsNumber** – The amount of segments in the message (See "Message character length").
- **OriginalMessage** – The content of the sent message.
- **NotificationDate** – Time of receiving the notification from the cellular operator.

Delivery Notification Example

URL: http://www.clientURL.co.il/demo.aspx

With parameters:
PhoneNumber=0527777888&Network=052&Status=2&StatusDescription=Delivered&ProjectId=10838&CustomerMessageId=652&CustomerParam=&id=&SenderNumber=4234324&BillingCodeId=1&Price=0.00&SegmentsNumber=2&ActionType=Content&OriginalMessage=test 1&NotificationDate=08/01/2017 10:12:07&RetriesNumber=0

Option 2 (PULL)

In order to pull notifications you need to enter the following address under the DeliveryNotificationUrl tag. http://uapi.inforu.co.il/InsertNotificationsToPullQueue.ashx

NOTE: Once pulled, the notification entries are removed from our system queue.

Web Service Address:
http://uapi.inforu.co.il/ClientServices.asmx

Function name: PullClientDLR

Parameters: Explanation of Functions

- **UserName** – Username of the account that was supplied by InforUMobile
- **Password** – Password of the account
- **batchSize** – The amount of notifications you wish to pull

Response XML

After you issue the request, an answer is returned in XML format:


```
<ClientNotification>
    <Status>OK</Status>
    <BatchSize>2</BatchSize>
    <Messages>
            <Message>
                    <Type>Notification </Type>
                    <PhoneNumber>0529999999</PhoneNumber>
                    <Network>052</Network>
                    <Status>2</Status>
                    <StatusDescription>Delivered</StatusDescription>
                    <CustomerMessageId>1234</CustomerMessageId>
                    <CustomerParam>A60</CustomerParam>
                    <SenderNumber>1315</SenderNumber>
                    <SegmentsNumber>1</SegmentsNumber>
                    <NotificationDate>29/11/2016 14:16:25</NotificationDate>
                    <SentMessage>test</SentMessage>
            </Message>
                    <Message>
                    <Type>Notification </Type>
                    <PhoneNumber>0546668887</PhoneNumber>
                    <Network>054</Network>
                    <Status>-2</Status>
                    <StatusDescription>Expired</StatusDescription>
                    <CustomerMessageId>1238</CustomerMessageId>
                    <CustomerParam>F321</CustomerParam>
                    <SenderNumber>1315</SenderNumber>
                    <SegmentsNumber>1</SegmentsNumber>
                    <NotificationDate>29/11/2016 14:16:25</NotificationDate>
                    <SentMessage>test</SentMessage>
            </Message>
    </Messages>
</ClientNotification>
```

If the system queue is empty, the response will look like so:

```
<ClientNotification>
    <Status>OK</Status>
    <BatchSize>0</BatchSize>
</ClientNotification>
```

An explanation of the return parameters:

Status (of the request) – Describes whether the request succeeded or not. The list of possible statuses can be found below.
BatchSize – The number of notifications transferred.
Type – Will always contain the word 'Notification'.
PhoneNumber – The phone number for which the notification was received.
Network – The destination network
Status (of the message) – Possible values: 2 (received), -2 (not received), -4 (blocked).
StatusDescription – Given reason in case of message not received.
CustomerMessageID – Message identity parameter which is transferred during the time the message is sent. This is helpful when matching notifications received with messages sent.
CustomerParam – An additional, optional parameter for message identity.
SenderNumber – Phone number of the sender defined in the sent message.
SegmentsNumber – The number of segments sent.
NotificationDate – Time of receiving the notification from the cellular operator.
SentMessage - The content of the sent message.

List of Possible Statuses

- OK
- Failed
- BadUserNameOrPassword

Receiving SMS (optional)

Option 1(PUSH)

When a user sends an SMS from his cell phone, the system will perform an HTTP Post Request to the client's predefined URL (via customer support). The client receives the XML in the “IncomingXML” parameter, encoded in UTF-8.

Receiving SMS Example

```
http://ClientWebApp.co.il/GetSMS.aspx?IncomingXML=
<IncomingData>
    <PhoneNumber>0509999111</PhoneNumber>
    <Keyword>Yes</Keyword>
    <Message> Yes I am interested </Message>
    <Network>054</Network>
    <ShortCode>039444666</ShortCode>
    <CustomerID>17</CustomerID>
    <ProjectID>456</ProjectID>
    <ApplicationID>33321</ApplicationID>
</IncomingData>
```

Your site needs to answer:

```
<Inforu>OK</Inforu>
```

An explanation of the return parameters:

- **PhoneNumber** – The phone number of the message.
- **Keyword** – First word in the message.
- **Message** – The message incoming message.
- **Network** – The source network.
- **ShortCode** – Phone number to which the incoming message was sent.
- **CustomerID** – Customer ID in our system.
- **ProjectID** – Project ID in our system.
- **ApplicationID** – Application ID in our system.

Option 2 (PULL)

Purpose of the service: Pulling incoming messages which were received from the operators are waiting in the DB under PULL instead of PUSH. In order to activate this option, refer to the support department which will enable it for you.

NOTE: Once pulled, the MO entries are removed from our system queue.

Format

Microsoft .Net users should add reference to this URL:

http://uapi.inforu.co.il/ClientServices.asmx

Java / Delphi / Magic users should use this URL:

http://uapi.inforu.co.il/ClientServices.asmx?wsdl

The function name is PullClientMO.

Parameters: Explanation of Functions

- **UserName** – Username of the account that was supplied by InforUMobile
- **Password** – Password of the account
- **batchSize** – The amount of MO you wish to pull

Response:

After you issue the request, an answer is returned in XML format:


```
<ClientNotification>
    <Status>OK</Status>
    <BatchSize>2</BatchSize>
    <Messages>
        <Message>
            <Type>MoMessage</Type>
            <PhoneNumber>0521111111</PhoneNumber>
            <Network>052</Network>
            <Status></Status>
            <StatusDescription></StatusDescription>
            <CustomerMessageId></CustomerMessageId>
            <CustomerParam></CustomerParam>
            <SenderNumber>1315</SenderNumber>
            <SegmentsNumber></SegmentsNumber>
            <NotificationDate>29/12/2017 13:10:25</NotificationDate>
            <SentMessage>test</SentMessage>
        </Message>
        <Message>
            <Type>MoMessage</Type>
            <PhoneNumber>0548889999</PhoneNumber>
            <Network>054</Network>
            <Status></Status>
            <StatusDescription></StatusDescription>
            <CustomerMessageId></CustomerMessageId>
            <CustomerParam></CustomerParam>
```

```
                <SenderNumber>0000</SenderNumber>
                <SegmentsNumber></SegmentsNumber>
                <NotificationDate>29/12/2017 14:16:26</NotificationDate>
                <SentMessage>test 222</SentMessage>
            </Message>
    </Messages>
</ClientNotification>
```


If the system queue is empty, the response will look like so:


```
<ClientNotification>
    <Status>OK</Status>
    <BatchSize>0</BatchSize>
</ClientNotification>
```


An explanation of the return parameters:

Status (of the request) – Describes whether the request succeeded or not. The list of possible statuses can be found below.
BatchSize – The number of MO transferred.
Type – Will always contain the word 'MoMessage'.
PhoneNumber – The number that sent the incoming message.
Network – The source network.
Status – Not in use.
StatusDescription – Not in use.
CustomerMessageID – Not in use.
CustomerParam – Not in use.
SenderNumber – Phone number to which the incoming message was sent.
SegmentsNumber – Not in use.
NotificationDate – Time of receiving the MO from the cellular operator.
SentMessage – The content of the message which was received by the respondent.

List of Possible Statuses

- OK
- Failed
- BadUserNameOrPassword

SMS Tools

Delete Future Messages

You can delete future sent messages programmatically using a WebService.

In order to delete these messages, one must input any data in the CustomerMessageID and CustomerParam parameters at the time in which the message was originally sent.

Format

Microsoft .Net users should add reference to this URL:

http://uapi.inforu.co.il/ClientServices.asmx

Java / Delphi / Magic users should use this URL:

http://uapi.inforu.co.il/ClientServices.asmx?wsdl

Function name: DeleteFutureMessage.

Parameters: Explanation of Functions

- **UserName** – Username of the account that was supplied by InforUMobile
- **Password** – Password of the account
- **PhoneNumber** – Number to which the message was sent
- **CustomerMessageId** – ID sent in sending the message
- **CustomerParam** – Parameter sent in sending the message

Response:

```
<ClientNotification>
    <Status>OK</Status>
    <BatchSize>1</BatchSize>
</ClientNotification>
```

List of Possible Statuses

- **Status**
  - OK – Success
  - Failed – The request failed
- **BatchSize** – Number of future messages deleted.

Notes

- Deletion will be available only for singular messages sent, and not group messages sent.
- If data was not inputted in the CustomerParam and CustomerMessageID parameters , the request will fail.

Remaining SMS Quota

A function that returns the amount of SMS messages remaining in the user account.

Format

Microsoft .Net users should add reference to this URL:

http://uapi.inforu.co.il/WebTools.asmx

Java / Delphi / Magic users should use this URL:

http://uapi.inforu.co.il/WebTools.asmx?wsdl

Function name: GetUserQuota

URL GET or POST:

http://uapi.inforu.co.il/WebTools/GetUserQuota.ashx

Parameters:

- **UserName** – Username of the account that was supplied by InforUMobile.
- **Password** – Password of the account.

An example of a request in HTTP Get:

http://uapi.inforu.co.il/WebTools/GetUserQuota.ashx?userName=XXX&password=XXX

Response:

```
<Result>
        <Status>1</Status>
        <Data>9955.00</Data>
</Result>
```

List of Possible Statuses

-26 - Error User ID, Data empty
1- OK, Data Contains the Quota

Phone Number block check

The function checks if a certain phone number is blocked for receiving SMS.

Format

Microsoft .Net users should add reference to this URL:

http://uapi.inforu.co.il/WebTools.asmx

Java / Delphi / Magic users should use this URL:

http://uapi.inforu.co.il/WebTools.asmx?wsdl

Function name: CheckIfNumberBlocked

URL GET or POST:

http://uapi.inforu.co.il/WebTools/CheckIfNumberBlocked.ashx

Parameters:

- **UserName** – Username of the account that was supplied by InforUMobile.
- **Password** – Password of the account.
- **PhoneNumber** – Phone number (string)
- **PrepaidAmount** – Not in use. (Use 0 always)

An example of a request in HTTP Get:

http://uapi.inforu.co.il/WebTools/CheckIfNumberBlocked.ashx?userName=XXX&password=XXX&phoneNumber=0529999999&prepaidAmount=0

Response:

After you issue the request, an answer is returned in XML format:

```
<Result>
        <Status>1</Status>
        <Data></Data>
</Result>
```

List of Possible Statuses

-26 - Error User ID, Data empty
-44 - No permission to access, Data empty
-49 - Phone number is blocked, Data empty
  1 - Phone number is open, Data empty

Blocking Phone Numbers

Blocking numbers from receiving messages programmatically using WebService or POST.

Format

Microsoft .Net users should add reference to this URL:

http://uapi.inforu.co.il/WebTools.asmx

Java / Delphi / Magic users should use this URL:

http://uapi.inforu.co.il/WebTools.asmx?wsdl

Function name: BlockPhoneNumbers

URL GET or POST:

http://uapi.inforu.co.il/WebTools/BlockNumbersMT.ashx

Parameters:

- **UserName** – Username of the account that was supplied by InforUMobile.
- **Password** – Password of the account.
- **PhonesList** – The list of numbers to be blocked. Separate using semicolon (;).

An example of a request in HTTP Get:

http://uapi.inforu.co.il/WebTools/BlockNumbersMT.ashx?UserName=david&Password=123456&PhonesList=0549999999;0556325698

Response:

After you issue the request, an answer is returned in XML format:

```
<Result>
    <Status>1</Status>
</Result>
```

List of Possible Statuses

1 – Success
-26 – User identitiy failed
-33 – No correct phone numbers were found

Contacts & Groups Management

Create or Update contact

This function create or update contacts. The key for update is email or phone or both.
If you create 5 contacts or less, the response will be synchronic and final, else the response contain RequestToken ,You can check back later what the result of your request was.

https://capi.inforu.co.il/api/Contact/Update?json=

Example simple

```
{
    "User": {
            "Username": " USER ",
            "Token": " TOKEN "
    },
    "Data": {
             "Contacts": [
                          {
                          "FirstName": "David",
                          "LastName": "Cohen",
                          "PhoneNumber": "0589995551",
                          "Email": "Cohen@test.com"
                           },
                           {
                          "PhoneNumber": "0589995552"
                           }
                         ]
    }
}
```

Response

```
{
   "StatusId": 1,
   "StatusDescription": "Success",
   "DetailDescription": "",
   "FunctionName": "api/Contact/Update",
   "RequestToken": "",
   "Records": 2,
   "ReturnData": null
}
```

Status:

- 1 = OK
- -1 = Failed

Example Advanced


```
{
    "User": {
            "Username": " USER ",
            "Token": " TOKEN "
    },
    "Data": {
             "Contacts": [
                          {
                           "ContactRefId": 100001,
                          "FirstName": "David",
                          "LastName": "Cohen",
                          "PhoneNumber": "0589995551",
                          "Email": "Cohen@test.com",
                          "FirebaseToken": "abcd1234",
                          "Platform": "ANDROID"
                          "GenderId": 1,
                          "BirthDate": "1980-07-23",
                          "AdditionalDate": "2012-05-02",
                          "RegistrationDate": "2015-05-10",
                          "AddToGroupNumbers": "1,2,3",
                          "RemoveFromGroupNumbers": "4,5,6",
                          "AddToGroupName": "New group name",
                          "HandleEvents": true,
                          "Text1": "",
                          "Text2": "",
                          "Text3": "",
                          "Text4": "",
                          "Text5": "",
                          "Text6": "",
                          "Text7": "",
                          "Text8": "",
                          "Text9": "",
                          "Text10": "",
                          "Text11": "",
                          "Text12": "",
                          "Text13": "",
                          "Text14": "",
                          "Text15": "",
                          "Text16": "",
                          "Text17": "",
                          "Text18": "",
                          "Text19": "",
                          "Text20": "",
                          "Number1": "",
                          "Number2": "",
                          "Number3": "",
                          "Number4": "",
                          "Date1": "",
                          "Date2": ""
                           }
                         ]
    }
}
```

Explanation of parameters:

- **UserName** – Username of the account that was supplied by InforUMobile
- **Token** – Token of the account
- **Contacts** – Email or PhoneNumber are mandatory all other fields are optional.
  - **ContactRefId** - Optional - a unique id of the contact in the calling system
  - **GenderId** - 1=Male , 2 = Female
  - **FirebaseToken** – use for send notifications to google devices and apple,
    if you send only FirebaseToken without Email or PhoneNumber, you should also send ContactRefId.
  - **Platform** – "ANDROID" / "IOS", If you use FirebaseToken you should also send Platform.

Check Status Token

This function checks the final response for your request (if you create more than 5 contacts)

https://capi.inforu.co.il/api/Contact/CheckJobStatusByToken?json=

```
{
    "User": {
            "Username": " USER ",
            "Token": " TOKEN "
    },
    "Data": {
       "RequestToken": "ljfkdsfds"
    }
}
```

Response

```
{
  "StatusId": 1,
  "StatusDescription": "Success",
  "DetailDescription": "",
  "FunctionName": "api/Contact/CheckJobStatusByToken",
  "RequestToken": "",
  "Records": 2,
  "ReturnData": null
}
```

1. **Status:**
   - 1 = OK
   - -1 = Failed
   - -4 = RequestToken not found
   - -5 = Job in progress
2. **StatusDescription –** Status interpretation.
3. **DetailDescription –** Detail Description.
4. **RequestToken –** Not in use.
5. **FunctionName –** The name of the function you To which turned
6. **Records –** Number of contacts created or updated
7. **ReturnData –** Not in use.