

Just160.com.au

Integration Guide

April 2007

1. Introduction

This guide explains the Just160 Web Service, as well as the CSV Batch File Import.

2. Web Service

The web service is a standard XML/SOAP web service. It was developed in .NET but can be consumed through any means that can access the service over SOAP, HTTP POST or HTTP GET. The service is accessible securely over HTTPS, but can also be accessed over HTTP.

Location

The web service is available at:

<https://just160.com.au/webservice/just160.asmx>

WSDL

The WSDL (web service description language) is available at:

<https://just160.com.au/webservice/just160.asmx?WSDL>

Available Functions

The functions available through the web service are:

- SendSMS
- RegisterNumber
- ActivateNumber

SendSMS is the function used to send a single SMS to a given number.

RegisterNumber is used to add additional "from" numbers. These are the numbers you can use as the message sender, and will appear on the recipient's phone as the number the message came from.

ActivateNumber is used to activate the numbers registered through the RegisterNumber function. After registering a number you will receive an activation code, which you must then use in the ActivateNumber function.

Hourly, Daily, Weekly and Monthly Limits

If you would like Just160 to place a limit on the number of messages that can be sent through your account in a given time period, please contact us. There is no extra charge for this.

The SendSMS function

This function is used to send a single SMS to a given number.

The parameters to this function are:

<u>Parameter</u>	<u>Description</u>	<u>Required?</u>	<u>Data Type</u>
clientID	Your client ID assigned to you by Just160. This is available on the main screen once you have logged into the website.	Yes	Numeric
password	The password you chose when you created your account	Yes	Text
clientKey	Your client key assigned to you by Just160. This is available on the main screen once you have logged into the website.	Yes	Text
fromNumber	The number you want to appear on the recipient's phone as the sender. You must have registered this number and activated it.	Yes	Numeric
toNumber	The number to send the message to. This should be formatted in full international format with the leading country code. For example, Australian number 0409964716 should be entered as 61409964716.	Yes	Numeric
textMessage	The message to send. Maximum length is 160 characters. Anything extra will be ignored.	Yes	Text

The response fields from this function are:

<u>Parameter</u>	<u>Description</u>	<u>Data Type</u>
successful	Indicates whether the message was accepted by the gateway or not. This does not indicate successful sending of the message – only acceptance of the message at the gateway.	Boolean
errorCode	If the message was unsuccessful, an error code.	Numeric
errorMessage	If the message was unsuccessful, an error description.	Text
messageID	If the message was successful, a unique message ID	Numeric

SOAP Definitions:

see <https://just160.com.au/webservice/just160.asmx?op=SendSMS>

HTTP GET Example:

<https://just160.com.au/webservice/just160.asmx/SendSMS?clientID=123&password=XXX&clientKey=XX&fromNumber=XXX&toNumber=XXX&textMessage=XXX>

HTTP POST:

Simply post a form to <https://just160.com.au/webservice/just160.asmx/SendSMS> with the fields as defined in the parameter table above

For more detailed examples, see the end of this document.

The RegisterNumber function

This function is used to register additional “from” numbers. These are the numbers you can use as the message sender, and will appear on the recipient’s phone as the number the message came from.

The parameters to this function are:

<u>Parameter</u>	<u>Description</u>	<u>Required?</u>	<u>Data Type</u>
clientID	Your client ID assigned to you by Just160. This is available on the main screen once you have logged into the website.	Yes	Numeric
Password	The password you chose when you created your account	Yes	Text
clientKey	Your client key assigned to you by Just160. This is available on the main screen once you have logged into the website.	Yes	Text
newFromNumber	The number to register. This should be formatted in full international format with the leading country code. For example, Australian number 0409964716 should be entered as 61409964716.	Yes	Numeric

SOAP Definitions:

see <https://just160.com.au/webservice/just160.asmx?op=RegisterNumber>

HTTP GET Example:

[https://just160.com.au/webservice/just160.asmx/
RegisterNumber?clientID=123&password=XXX&clientKey=XXX&newFromNumber=XXX](https://just160.com.au/webservice/just160.asmx/RegisterNumber?clientID=123&password=XXX&clientKey=XXX&newFromNumber=XXX)

HTTP POST:

Simply post a form to <https://just160.com.au/webservice/just160.asmx/RegisterNumber> with the fields as defined in the parameter table above

The ActivateNumber function

This function is used to activate the numbers registered through the RegisterNumber function. After registering a number you will receive an activation code (via SMS), which you must then use pass into this function.

The parameters to this function are:

<u>Parameter</u>	<u>Description</u>	<u>Required?</u>	<u>Data Type</u>
clientID	Your client ID assigned to you by Just160. This is available on the main screen once you have logged into the website.	Yes	Numeric
password	The password you chose when you created your account	Yes	Text
clientKey	Your client key assigned to you by Just160. This is available on the main screen once you have logged into the website.	Yes	Text
fromNumber	The number you are activating. This must match exactly with the number registered through the RegisterNumber function.	Yes	Numeric
activationCode	The code received via SMS to the number you are activating.	Yes	Text

SOAP Definitions:

see <https://just160.com.au/webservice/just160.asmx?op=ActivateNumber>

HTTP GET Example:

<https://just160.com.au/webservice/just160.asmx/ActivateNumber?clientID=123&password=XXX&clientKey=XXX&fromNumber=XXX&activationCode=XX>
[X](#)

HTTP POST:

Simply post a form to <https://just160.com.au/webservice/just160.asmx/ActivateNumber> with the fields as defined in the parameter table above

ASP.NET Example

This example assumes you are using Visual Studio .NET as your development environment.

Once you have opened (or created) your project, right-click the project in the solution explorer and select "Add Web Reference". In the window that pops up, enter <https://just160.com.au/webservice/just160.asmx> in the URL box and click GO.

Once Visual Studio finds the service it will display the description, and place "au.com.just160" in the Web Reference Name box on the right. Click the "Add Reference" button to add the reference to your project.

You can now use the web service from any function in your project. Place the following code where you want to send a message:

C#

```
// create an instance of the proxy class
au.com.just160.Just160Message myMessage;
myMessage = new au.com.just160.Just160Message();

// create the object to hold the response
au.com.just160.Just160Response myResponse;

// call the SendSMS function
myResponse = myMessage.SendSMS(123,"xxx","xxx","61409964716",
                               "61409964716", "test message");

// check whether the message was successful
if (myResponse.successful)
{
    // do something to indicate success to the user
    // logging of myResponse.messageID is useful to track messages
}
else
{
    // do something to indicate failure to the user
    // use myResponse.errorCode and myResponse.errorMessage to
    // determine the reason for the failure
}
```

VB.NET

```
` create an instance of the proxy class
Dim myMessage as au.com.just160.Just160Message
Set myMessage = New au.com.just160.Just160Message()

` create the object to hold the response
Dim myResponse as au.com.just160.Just160Response

` call the SendSMS function
myResponse = myMessage.SendSMS(123,"xxx","xxx","61409964716",
                               "61409964716", "test message")

` check whether the message was successful
If myResponse.successful Then
    ` do something to indicate success to the user
    ` logging of myResponse.messageID is useful to track messages
Else
    ` do something to indicate failure to the user
    ` use myResponse.errorCode and myResponse.errorMessage to
    ` determine the reason for the failure
End If
```

HTML / Javascript Example

Placing the following code into a HTML page will create a web page that can send an SMS message through our web service.

```
<html>
<head>
<script language="Javascript">
    function xmlhttpPost(strURL)
    {
        var xmlhttpReq = false;
        var self = this;

        if (window.XMLHttpRequest)
        {
            self.xmlHttpRequest = new XMLHttpRequest();
        }
        else if (window.ActiveXObject)
        {
            self.xmlHttpRequest = new ActiveXObject("Microsoft.XMLHTTP");
        }

        self.xmlHttpRequest.open('POST', strURL, true);
        self.xmlHttpRequest.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded');
        self.xmlHttpRequest.onreadystatechange =
            function()
            {
                if (self.xmlHttpRequest.readyState == 4)
                {
                    updatepage(self.xmlHttpRequest.responseText);
                }
            }

        self.xmlHttpRequest.send(getQueryString());
    }

    function getQueryString()
    {
        var form = document.forms['form1'];
        qstr = 'toNumber=' + escape(form.toNumber.value)
            + '&textMessage=' + escape(form.textMessage.value)
            + '&clientID=' + escape(form.clientID.value)
            + '&password=' + escape(form.password.value)
            + '&clientKey=' + escape(form.clientKey.value)
            + '&fromNumber=' + escape(form.fromNumber.value);
        return qstr;
    }

    function updatepage(str)
    {
        if (str.indexOf('<successful>true</successful>') > -1)
            alert('Message delivered to gateway successfully');
        else
            alert('Failed to send message');
    }
</script>
</head>

<form name="form1">
    <p>
        To Number: <input name="toNumber" type="text"><br>
        Message: <input name="textMessage" type="text"><br>
        <input name="clientID" type="hidden" value="123">
        <input name="password" type="hidden" value="xxx">
        <input name="clientKey" type="hidden" value="xxx">
        <input name="fromNumber" type="hidden" value="61409964716">
        <input value="Go" type="button"
            onclick="JavaScript:xmlhttpPost('https://just160.com.au/webservice/just160.asmx/SendSMS') '>
    </p>
</form>

</body>
</html>
```

Error Codes

The following error codes can be returned by the web service:

100	Invalid Client Details
200	Account Not Activated
300	From Number Not Activated
310	From Number Already Exists
320	Invalid Activation Code
400	Invalid Data Type
700	Hourly Limit Exceeded
701	Daily Limit Exceeded
702	Weekly Limit Exceeded
703	Monthly Limit Exceeded
800	Not Enough Credit
810	All Free Messages Have Been Used
900	Internal Error

3. CSV Batch File Import

If you are processing large batches of messages, you may import a Comma-Separated-Values (CSV) file containing your messages.

Once logged in to our website, click the “Batch File Upload” menu option from the left menu,

The CSV file must contain 3 columns:

- From Phone Number
- To Phone Number
- Message

For example, a file could contain:

```
61400123456,61411234567,This is a test message
61400123456,61422345678,This is a test message
```

From the import screen, select your file and click the “Import File” button.

The system will then scan your file for errors and alert you of any problems. Assuming all is correct, you will see the list of messages you are about to send. Review them and click the “Send Messages” button to send your messages.

Finally, the system will show the results of processing your messages.

NOTE: messages sent through the batch import process will be charged at your normal “per-message” rate