



Open API Project

OneAPI v2.0 SMS REST

Document Version v1.1

Document Revision History

Rev #	Date	Description
1.0	June 20 2013	Initial version, based on Aepona OneAPI2.0 SMS REST API Guide doc v1.1h
1.1	Nov 1 2013	Updated URI, examples to match production values. Added information on sendSms Sandbox service.

Copyright

2013

© Aepona Limited,

Beacon House,

Clarendon Dock,

Belfast,

BT1 3BG

All rights reserved. This document or any part thereof may not, without the written consent of Aepona Limited, be copied, reprinted or reproduced in any material form including but not limited to photocopying, transcribing, transmitting or storing it in any medium or translating it into any language, in any form or by any means, be it electronic, mechanical, xerographic, optical, magnetic or otherwise.

The information contained in this document is proprietary and confidential and all copyright, trademarks, trade names, patents and other intellectual property rights in the documentation are the exclusive property of Aepona Limited unless otherwise specified. The information (including but not limited to data, drawings, specification, documentation, software listings, source or object code) shall not at any time be disclosed directly or indirectly to any third party without Aepona Limited's prior written consent.

The information contained herein is believed to be accurate and reliable. Aepona Limited accepts no responsibility for its use by any means or in any way whatsoever. Aepona Limited shall not be liable for any expenses, costs by damage that may result from the use of the information contained within this document. The information contained herein is subject to change without notice.

Table of Contents

1	SMS REST Overview	6
2	Authentication	6
3	Methods	6
3.1	URIs	7
3.2	Parameters	8
4	Sending SMS	8
4.1	Send an SMS from Your Application	8
4.1.1	Request	8
4.1.2	Request Parameters	9
4.1.3	Response	10
4.1.4	Response Parameters	10
4.2	Query the Delivery Status of an SMS	11
4.2.1	Request	11
4.2.2	Request Parameters	11
4.2.3	Response	11
4.2.4	Response Parameters	12
4.3	Notify Client About Message Delivery Status	13
4.3.1	Request	13
4.3.2	Request Parameter	13
4.3.3	Response	14
4.3.4	Response Parameters	14
5	Receiving SMS	14
5.1	Retrieve SMS Sent to your Application	14
5.1.1	Request	14
5.1.2	Request Parameters	15
5.1.3	Response	15
5.1.4	Response Parameters	16

5.2	Notify Client about Message Arrival.....	17
5.2.1	Response.....	18
5.2.2	Response Parameters.....	18
6	Response Codes & Exceptions.....	18
6.1	Response Codes.....	18
6.2	Exceptions.....	19
6.2.1	Service Exceptions.....	19
6.2.2	Policy Exceptions.....	20
A	Starting/Stopping Notifications.....	22
A.1	Starting Notifications.....	22
A.1.1	Requesting Codes.....	22
A.1.3	Creating Notifications.....	25
B	Pausing, Restarting and Stopping Notifications.....	26
C	De-assigning and Reactivating Keywords.....	27

1 SMS REST Overview

The SMS interface allows an application to send and receive SMS messages. You can find some examples of when you may want to do this in the use cases at <http://www.oneapi.gsmworld.com>.

An SMS sandbox service is also provided, URI and examples for which are contained below and in Appendix D, and details on how to configure the service in a separate SandboxDataService API guide.

! Throughout this document, the examples may be shown WITHOUT URL encoding for readability purposes, e.g. if the address "tel:+12345678900" is in the URL example, this should be encoded as "tel%3A%2B12345678900", where the character ":" is "%3A" and the character "+" is "%2B"

2 Authentication

A server side certificate is required plus HTTP Basic Authentication.

For more information, refer to the 'Developer Access' section in the 'OneAPI v2.0 Common Information Guide'.

3 Methods

The following methods are available:

- Sending SMS:
 - Send an SMS from Your Application
 - Query the Delivery Status of an SMS
 - Notify Client About Message Delivery Status
- Receiving SMS:
 - Retrieve SMS Sent to your Application (which is identified by **registrationId**)
 - Notify Client about Message Arrival

POST and GET are used in OneAPI SMS.

Steps to start and stop notifications are described in Appendix A.

? The SMS API supports JSON content types for POST operations. The response content type is application/JSON.

3.1 URIs

The URIs of the resources are as follows:

- Sending SMS:

Send an SMS from your Application (to one or more mobile terminals) – described in section 4.1.

<https://developerportal.uscellular.com/services/smsSend/{api version}/smsmessaging/outbound/{senderAddress}/requests>

Query the delivery status of an SMS (identified by requestId) – described in section 4.2.

<https://developerportal.uscellular.com/services/smsSend/{api version}/smsmessaging/outbound/{senderAddress}/requests/{requestId}/deliveryInfos>

A message delivery status notification is sent to the client by the server, with the format as described in section 4.3

- Sending SMS from your Application to the Sandbox smsSend service – described in Appendix D:

<https://developerportal.uscellular.com/services/smsSendSandbox/{api version}/smsmessaging/outbound/{senderAddress}/requests>

- Receiving SMS:

Retrieve SMS sent to your application (which is identified by **registrationId**) – described in 5.1

<https://developerportal.uscellular.com/services/receiveSms/{api version}/smsmessaging/inbound/registrations/{registrationId}/messages>

A message delivery status notification is sent to the client by the server, with the format as described in section 5.2.

The variables used in request URLs are described below:

Name	Description
{apiVersion}	Version of the API that the client wants to use. In this case 2_0.
{senderAddress}	Typically the SMS short code, which identifies the client application. If this is used as a parameter in the request body, the values must match.
{requestId}	The outbound message request ID generated by the server.
{registrationId}	The reference to the off-line retrieval criteria provisioned in advance and known to the application.

! The SMS API supports JSON content types for POST operations. The response content type is application/JSON.

3.2 Parameters

Inline parameters are contained in defined data structures, which are noted by their Type. For example, in the Request sample code at section 4.1.1 below, the parameters are contained within the data structure *outboundSMSMessageRequest*. Many data structures nest other sub-structures.

4 Sending SMS

4.1 Send an SMS from Your Application

This allows you to send an SMS from your application to one or more addresses.

4.1.1 Request

```
POST
https://developerportal.uscellular.com/services/smsSend/2_0/smsmessaging/outbound/12345/re
quests HTTP/1.1
Content-Type: application/json
Authorization: Basic QWVwb25hVfdghfdghdfghdfgWF6eHN3MjNI
```



```

Content-Length: 328
Host: developerportal.uscellular.com
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
{"outboundSMSMessageRequest":
  {
    "address": ["tel:+14567800001"],
    "clientCorrelator": "105698",
    "outboundSMSTextMessage": {"message": "Test Message"},
    "receiptRequest": {"notifyURL": "http://example.com/MockRestService/rest"},
    "senderAddress": "12345",
    "senderName": "Aepona"
  }
}

```

4.1.2 Request Parameters

Table 1: Send an SMS - Request Parameters (outboundSMSMessageRequest Type)

! Use POST to create the SMS. The **senderAddress** in the URL must be URL-escaped.

Parameter	Data Type	Description	Optional
address	xsd:anyURI [1..unbounded]	Destination addresses for the message. Address (CTN) of U.S. Cellular subscriber, in the format comprising: <ol style="list-style-type: none"> 1) 'tel:' protocol identifier, 2) country code of one preceded by '+', 	No

		3) ten digit CTN preceded by 1, for example, tel:+15087300001	
clientCorrelator	xsd:string	<p>A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server.</p> <p>This field SHOULD be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids re-sending the message in such situations.</p>	Yes
outboundSMSTextMessage	outboundSMSTextMessage	Contains the message parameter with the associated message text.	No
receiptRequest	common: CallbackReference	The application endpoint that will be used to notify the application when the message has been delivered to a terminal or delivery is impossible. The parameter includes the URL-escaped notifyURL.	Yes
senderAddress	xsd:anyURI	The address of the sender to whom a responding message may be sent. This value should be set to the short code assigned to your application.	No
senderName	xsd:string	Name of the sender to appear on the user's terminal as the originator of the message.	Yes

4.1.3 Response

```

HTTP/1.1 201 Created
Date: Thu, 31 Oct 2013 11:31:43 GMT
Server: Jetty(6.1.x)
Content-Type: application/json
Host: developerportal.uscellular.com

```

```

Authorization: Basic QWVwb25hVfdghfdghdfghdfgWF6eHN3MjNI
breadcrumbId: ID-ase1-47049-1383117280034-17-14
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
Location:
http://developerportal.uscellular.com/services/smsSend/2_0/smsmessaging/outbound/12345/req
uests/124400101
X-Forwarded-Host: developerportal.uscellular.com
Content-Length: 160
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
{
  "resourceReference" : {
    "resourceURL" :
"http://developerportal.uscellular.com/services/smsSend/2_0/smsmessaging/outbound/12345/re
quests/124400101"
  }
}

```

201 shows that the message was created.

4.1.4 Response Parameters

The Location header field shows the URI of the created message, including the **senderAddress** and **requestID** in the path. You can append **'/deliveryInfos'** to this URI to query the delivery status of an SMS as described in the next section 4.2.

Table 2: Send an SMS - Response Parameters

Parameter	Data Type	Description	Optional
resourceURL	xsd:anyURI	Self referring URL.	No

This URI is also included in the response body as the **resourceURL** pair within the **resourceReference** object.

4.2 Query the Delivery Status of an SMS

You can query the delivery status of an SMS which has been sent from your application.

4.2.1 Request

```
GET
https://developerportal.uscellular.com/services/smsSend/2_0/smsmessaging/outbound/12345/requests/1244
00101/deliveryInfos HTTP/1.1

Authorization: Basic QWVwb25hVfdghfdghdfghdfgWF6eHN3MjNI

Host: developerportal.uscellular.com

Connection: Keep-Alive

User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

4.2.2 Request Parameters

Table 3: Query the Delivery Status - Request Parameters

Parameter	Data Type	Description	Optional
requestId	URI variable	This identifies the specific SMS delivery request. In the examples, this value is '60022', which was returned when the message was created. (See Send an SMS from Your Application on page 8.)	No

4.2.3 Response

```
HTTP/1.1 200 OK

Date: Thu, 31 Oct 2013 11:43:05 GMT

Server: Jetty(6.1.x)

Content-Type: application/json

Authorization: Basic QWVwb25hVfdghfdghdfghdfgWF6eHN3MjNI

breadcrumbId: ID-ase3-43542-1383117279982-22-16

Host: developerportal.uscellular.com
```

```

User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
X-Forwarded-Server: developerportal.uscellular.com
Content-Length: 291
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
{
  "deliveryInfoList" : {
    "resourceURL" :
    "http://developerportal.uscellular.com/services/smsSend/2_0/smsmessaging/outbound/12345/re
quests/124400101/deliveryInfos",
    "deliveryInfo" : [ {
      "address" : "tel:+14567800001",
      "deliveryStatus" : "DeliveredToTerminal"
    } ]
  }
}

```

4.2.4 Response Parameters

The deliveryInfoList type contains the delivery information for each address to which you sent the message, in a **deliveryInfo** array.

Table 4: Query the Delivery Status - Response Parameters (deliveryInfoList Type)

Parameter	Data Type	Description	Optional
resourceURL	xsd:anyURI	This is a reference to this response.	No
deliveryInfo	deliveryInfo [1... unbounded]	Contains delivery information. (See table 4.1 for details).	No

Table 4.1: Query the Delivery Status - Response Parameters (deliveryInfo Type)

Parameter	Data Type	Description	Option al
-----------	-----------	-------------	-----------

address	xsd:anyURI	Outbound message destination address.	No
deliveryStatus	DeliveryStatus	Contains the delivery result for the destination address. (See table 4.2 for details).	No

Table 4.2: DeliveryStatus

Status	Description
DeliveredToTerminal	Successful delivery to terminal.
DeliveryUncertain	Delivery status unknown, e.g. because it was handed off to another network.
DeliveryImpossible	Unsuccessful delivery; the message could not be delivered before it expired.
MessageWaiting	The message is still queued for delivery. This is a temporary state, pending transition to one of the preceding states.
DeliveredToNetwork	Successful delivery to the network enabler responsible for routing the SMS.

4.3 Notify Client About Message Delivery Status

This section describes the format of delivery notifications sent from the server to the client application. A delivery notification is sent to the client application callback URL provided in the sendSMS request, typically when the message is DeliveredToTerminal or DeliveryImpossible.

Starting and stopping notifications is carried out via the Dev Portal. For instructions on how to do this, please see Appendix A.

4.3.1 Request

```
POST /notifications/DeliveryInfoNotification HTTP/1.1
Accept: application/json
Content-Type: application/json
Content-Length: nnnn
Host: application.example.com
{"deliveryInfoNotification": {
```

```

    "deliveryInfo": {
      "address": "tel:+15555550101",
      "deliveryStatus": "DeliveredToNetwork"
    }
  }
}

```

4.3.2 Request Parameter

**Table 5: Notify Client about Message Delivery Status - Request Parameters
(deliveryInfoNotification Type)**

Parameter	Data Type	Description	Optional
deliveryInfo	deliveryInfo[1 ...unbounded]	Contains delivery information. (See table 5.1 for details).	No

**Table 5.1: Notify Client about Message Delivery Status - Request Parameter s
(deliveryInfo Type)**

Parameter	Data Type	Description	Optional
address	xsd:anyURI	Outbound message destination address. This would be the address (CTN) of U.S. Cellular subscriber, in the format comprising: <ol style="list-style-type: none"> 1) 'tel:' protocol identifier, 2) country code of one preceded by '+', 3) ten digit CTN preceded by 1, for example, tel:+15087300001 	No
deliveryStatus	DeliveryStatus	Contains the delivery result for the destination address. (See table 4.2 for details).	No

4.3.3 Response

HTTP/1.1 204 No Content

Date: Thu, 04 Jun 2009 02:51:59 GMT

4.3.4 Response Parameters

N/A

5 Receiving SMS

5.1 Retrieve SMS Sent to your Application

This method allows you to retrieve any SMS that have been sent to your application.

5.1.1 Request

GET

https://developerportal.uscellular.com/services/receiveSms/2_0/smsmessaging/inbound/3456abc/messages?maxBatchSize=2 HTTP/1.1

Authorization: Basic QWVwb25hVfdghfdghdfghdFGWF6eHN3MjNI

Host: developerportal.uscellular.com

5.1.2 Request Parameters

Table 6: Retrieve Messages Sent to your Application - Request Parameters

Parameter	Data Type	Description	Optional
registrationId	URI variable	This is the registration ID agreed with the OneAPI operator. In the sample code above, this value is '3456abc'.	No
maxBatchSize	URI variable	This is the maximum number of messages to retrieve in this request.	Yes

5.1.3 Response


```

HTTP/1.1 200 OK
Content-Type: application/json
{"inboundSMSMessageList": {
  "inboundSMSMessage": [ {
    "destinationAddress": "12345",
    "senderAddress": "tel:+14567800001",
    "message": "sampleMessage",
    "dateTime": "2013-10-22T15:43:33.000+08:00",
    "resourceURL":
    "https://developerportal.uscellular.com/services/receiveSms/2_0/smsmessaging/inbound/registra
tions/1100011/messages/1100017",
    "messageId": "1100017"
  }],
  "totalNumberOfPendingMessages": 0,
  "numberOfMessagesInThisBatch": 1,
  "resourceURL":
  "https://developerportal.uscellular.com/services/receiveSms/2_0/smsmessaging/inbound/registra
tions/1100011/messages "
}}

```

5.1.4 Response Parameters

Parameters are returned in the **inboundSMSMessageList** object.

Table 7: Retrieve Messages Sent to your Application - Response Parameters
(*inboundSMSMessageList* Type)

Parameter	Data Type	Description	Optional
inboundSMSMessage	inboundSMSMessage [0..unbounded]	Contains an array of messages received according to the specified registrationId. (See table 7.1 for details).	Yes
numberOfMessagesInThisBatch	xsd:int	The number of messages included in the response (part of the totalNumberOfPendingMessages).	Yes

resourceURL	xsd:anyURI	Self referring URL.	No
totalNumberOfPendingMessages	xsd:int	Total number of messages in the gateway storage waiting for retrieval at the time of the request.	Yes

Table 7.1: Retrieve Messages Sent to your Application - Response Parameters (inboundSMSMessage Type)

Parameter	Data Type	Description	Optional
dateTime	xsd:dateTime	The date and time at which the message was received.	Yes
destinationAddress	xsd:anyURI	This is the short code associated with the service.	No
messageId	xsd:string	This is a server-generated message identifier.	Yes
message	xsd:string	This is the SMS message itself.	No
resourceURL	xsd:anyURI	Self referring URL.	Yes
senderAddress	xsd:anyURI	This is the MSISDN of the sender.	No

5.2 Notify Client about Message Arrival

This section describes the format of SMS message arrival notifications sent from the server to the client application callback URL. The callback URL is configured for a given set of notification criteria (shortcode and keyword) in the Dev Portal.

Starting and stopping notifications is carried out via the Dev Portal. For instructions on how to do this, please see Appendix A.

The format of the message receipt notification sent to your **notifyURL** is:

```
POST /notifications/SMSNotification HTTP/1.1
Accept: application/json
Content-Type: application/json
Content-Length: nnnn
Host: application.example.com
```

```

{"inboundSMSMessageNotification": {
  "inboundSMSMessage": {
    "dateTime": "2009-11-19T12:00:00",
    "destinationAddress": "tel:+12345",
    "message": "First simple message",
    "messageId": "msg001",
    "senderAddress": "tel:+15555550120"
  }
}}

```

There will be one notification for every SMS received matching the notification criteria.

The **inboundMessageNotification** object includes an **inboundSMSMessage** array with the following parameters:

Table 8: Notification of a Received Message - Response Parameters
(*inboundSMSMessage Type*)

Parameter	Data Type	Description	Optional
dateTime	xsd:dateTime	The date and time at which the message was received.	Yes
destinationAddress	xsd:anyURI	This is the number associated with your service, e.g. an agreed short code.	No
messageId	xsd:string	This is a server-generated message identifier.	Yes
message	xsd:string	This is the SMS message itself.	No
resourceURL	xsd:anyURI	This is a link to the message.	Yes
senderAddress	xsd:anyURI	This is the MSISDN of the sender.	No

5.2.1 Response

The client application should return HTTP 204 – No Content.

```
HTTP/1.1 204 No Content
```

```
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

5.2.2 Response Parameters

N/A

6 Response Codes & Exceptions

6.1 Response Codes

HTTP response codes are used to indicate:

- **200** – Success!
- **400** – Bad request; check the error message for details
- **401** – Authentication failure, check your authentication details.
- **403** – Forbidden; please provide authentication credentials
- **404** – Not found: mistake in the host or path of the service URI
- **405** – Method not supported: for example you mistakenly used a HTTP GET to create an MMS instead of a POST
- **500** – Internal Error 500; the server encountered an unexpected condition which prevented it from fulfilling the request
- **503** – Server busy and service unavailable. Please retry the request.

For more details on these, refer to <http://www.ietf.org/rfc/rfc2616.txt>.

6.2 Exceptions

```
HTTP/1.1 400 Bad Request
```

```
Content-Type: application/json
```

```
Content-Length: 1234
```

```
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

```
{"requestError": {
```

```

"serviceError": {
  "messageId": "SVC0002",
  "text": " Invalid input value for message part %1",
  "variables": " tel:+016309700000"
}
}}

```

This section lists the available error codes, the possible reasons why the exception may have occurred, and possible solutions.

6.2.1 Service Exceptions

A service exception describes the reason why the service cannot accept the request.

The following service exceptions may be thrown:

Table 9: Service Error Codes

Error	Explanation
SVC0001 – Service error occurred	A service-related error has occurred as a result of a client invocation on the service. This category can be used for implementation-specific errors. Contact the support team.
SVC0002 – Invalid input value	An input parameter value is not of the expected type. Check the parameter types and re-submit your request.
SVC0004 – No valid address(es)	The requested terminal device address does not exist. Use an address that exists.

6.2.2 Policy Exceptions

A policy exception means that the request syntax is valid, however an operator policy has been broken.

POL0001 - Policy error occurred

The above exception may be thrown to indicate a fault relating to a policy associated with the service. This category can be used for implementation-specific errors such as:

Table 10: Policy Error Codes

Error	Explanation
POL-006: TPA exceeded its maximum allowed rate of transactions	The maximum rate of transactions is exceeded. Ensure that the rate of your requests is within the limits set up in your SLA, e.g. 10 TPS (Transactions Per Second).
POL-008: TPA is invalid	The Third Party Application authentication details are incorrect. Check your basic authentication username and password are correct and re-submit your request.
POL-014: White List is enforced, and address is not in White List	A white list is enforced and the number is not in the white list. Check your SLA details.
POL-015: Black List is enforced, and address is in Black List	A black list is enforced and the number is in the black list. Check you SLA details.
POL-016: Max Requests is enforced, and max requests has been exceeded	The maximum number of requests for this service is exceeded. Contact the support team.
POL-017: Operation is not allowed	The method/operation is not supported in your current SLA. Check your SLA and use a method that is supported.
POL-018: All targets were rejected for MDN access and authorization failure	<p>This indicates that none of the destination numbers can be retrieved by the internal address resolver such as LDAP or Lookup.</p> <p>It includes white/black list rejection when the destination number cannot be found in either list that is enforced. In this case, check your policy contract and request the number to be added to/removed from the appropriate list.</p>
POL-020: Max Message Length is enforced, and max message length has been exceeded	A maximum message length policy is in place and you have exceeded this. Check you SLA for the maximum message length, update your message and re-submit your request.

POL-021: Min Message Length is enforced, and message length is less than min allowed	A minimum message length policy is in place and you have a message length that is less than this minimum. Check your SLA for the maximum message length, update your message and re-submit your request.
POL-022: Receipting is enforced, and receipting has not been enabled	A receipt has been requested but it is not enabled for this service. Remove the receipt request and re-submit your request.
POL-040: Max Destination Addresses is enforced and maximum destination addresses has been exceeded	A maximum destination address limit is enforced and it has been exceeded. Check your SLA for the limit and re-submit your request.
POL-049: SPID Black List is enforced and address SPID is in the SPID Black List.	Applicable in multiple carrier deployments, Black List is enforced and the carrier identified by the Service Provider ID is in the black list. Therefore all the addresses from the carrier are rejected.

A Starting/Stopping Notifications

Starting and stopping notifications is managed from the Dev Portal. You will use one of the codes assigned to your Partner entity on the platform, or configure a callback URL, and keyword to automatically start receiving SMS arrival and receipt notifications. You can pause or stop by a button from the GUI.

The steps described below assume that you have a login on the Dev Portal which allows you to manage notifications for your Partner's applications.

A.1 Starting Notifications

The task is divided into three sub-sections in a standard workflow:

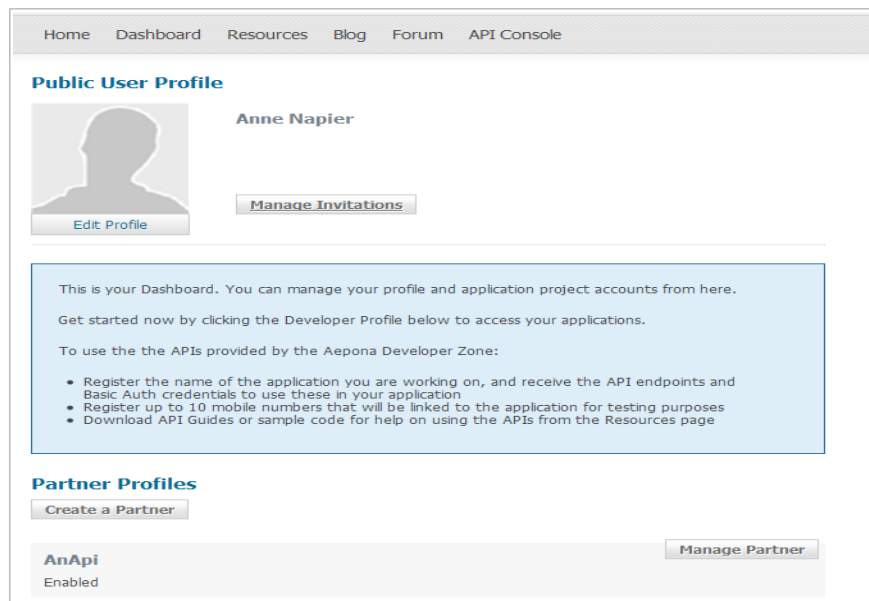
- 1 Requesting Codes - section below
- 2 Creating Keywords - section A.1.2
- 3 Creating Notifications - section A.1.3

A.1.1 Requesting Codes

You need to request the administrator to assign a code to your project.

- 1 Login to the Dev Portal.

The Dashboard is displayed, as shown below. Partners associated with you are displayed in the Partner Profiles area.



- 2 Click **Manage Partner** for the Partner whose application you want to set up the notification for.

The Partner Profile page is displayed. Registered applications are listed as projects in the Partner Projects area.

- 3 Click **Manage Project** for the relevant project.

The Project Profile page is displayed for your selected project.

- 4 Click **Manage Notifications**.

The Manage Notifications page is displayed.

- 5 Click **Request Number**.

The Create a New Number page is displayed.

- 6 Select the **Number Type** to request from the dropdown list: Short Code or Long Number.

- 7 Select to have a number assigned automatically by the Admin Portal administrator.

OR

to specify one for the project, by entering the **Number** in the field that will appear. This will be unique to your project, called an unshareable number.

- 8 Select the **Enabler Type**: SMS.


- 9 Select **Country** and **Networks** from the list displayed. You can assign only one network to a long number.



? If you are testing your application against the Sandbox version of the service, choose the Test network.

10 Click **Submit**.

Your request is submitted to the administrator, and until approved, will be listed in the Pending Requests page. (Click **Pending Requests** on the Manage Notifications page.)

You will receive an email when your request is approved. It will contain the number assigned. Your Manage Notifications page will show the assigned number in the Number panel as shown below:

Number	Type	Network(s)	Enabler Type(s)	
998877	Short Code	sandbox	SMS	

Showing 1 to 1 of 1 entries  

? You can remove the number from your project by using the red icon in the Number panel. To add it back, use the **Assign Partner Number** button.

? As an option, you can create **Callback Credentials** from the Manage Notifications page. This will add authentication for the notification to access your application.

A.1.2 Creating Keywords

You need to create a keyword in combination with the number you are going to use for your notification. Each keyword+number pair must be unique across the platform, which means that if you are creating a single notification using your application specific number, keyword isn't necessary. In this case you can skip the steps below and go directly to create your notification.

? A keyword can be created during notification creation, also.

- 1 From the Manage Notifications page, click **Create Keyword**. The Create Keyword page is displayed.
- 2 Specify the number you wish to associate the keyword with, from the **Number** dropdown list.

- 3 Enter the **Keyword**. An error message will be returned if the keyword is in use or it does not conform to the system rules.

Manage Notifications page is refreshed and shows the keyword created, in RESERVED status, as shown below:

Keywords	Number	Status	Expires
ppppppp	998877	RESERVED	N/A

Showing 1 to 1 of 1 entries

The status will change to ACTIVE when the keyword is in use with a notification. The RESERVED status will expire after a system configured number of days, for example, 180 days. See Appendix A3 .

A.1.3 Creating Notifications

You are now ready to create a notification, which will start immediately.

- 1 From the Manage Notifications page, click **Create Notifications**.
- 2 Select the **Notification Type** from the dropdown list. SMSX is available by default.
- 3 Select the **Number** to use from the dropdown list.
- 4 Select the **Protocol** from the dropdown list.

! If you are testing your application against the Sandbox version of the service, choose Sandbox-OneAPI_REST-v2_0.

- 5 Select the **Keyword** from the dropdown list, OR create one by clicking **Request Keywords**.
- 6 Select either Poll or Push notification model:

Notification Model	Description
Polling Message Retrieval	Cache & collect model. No application callback is required. The application will periodically retrieve messages from the platform via the API.
Push Notification	Direct push of notifications. An application callback is required.

- 7 If Push Notification is selected, enter the **Callback Url** in the field that appears:

Create Notification

Notification Type * SMSX
 Number * 23456
 Protocol * Sandbox-OneAPI_REST_v2_0

Keyword * yes

Polling Message Retrieval
 Push Notification

Callback Url *

or

8 Click **Save**.

The notification is started.




The Manage Notifications page will be refreshed to display the notification with its ID, and the Callback URL as entered, with the pause and stop buttons at the right hand end of the row, as shown in the screenshot below:

Notification ID	Number	Keyword	Notification Type	Protocol	Callback URL	
800000	998877	PPPPPPP	SMSX	UI_SMPP1-OneAPI_SOAP_v1_0	http://www.w3.org/2005/08/addressing/none	⏸ ⏹

Showing 1 to 1 of 1 entries

B Pausing, Restarting and Stopping Notifications

To pause, restart and stop a notification:


- 1 From the Dashboard, click **Manage Partner > Manage Project > Manage Notifications** to find the notification you wish to manage.
- 2 Click the appropriate button at the right end of the row:
 - To pause a notification that has been started, click the  icon.
 - To restart a notification that has been paused, click the  icon.
 - To stop a notification, click the  icon at the far right.
- 3 Confirm your action when prompted.

! When you stop a notification, it will be removed from the system. You will not be able to restart it, but you can recreate an identical notification.




C De-assigning and Reactivating Keywords

Keywords in RESERVED status can be de-assigned from the application and put into QUARANTINED status, and reactivated back to RESERVED status.

To de-assign a keyword from a project:

- 1 From the Manage Notifications page, find the keyword you wish to de-assign.
- 2 Click the  icon on the right.

The keyword is now in QUARANTINED status, as shown below, showing its expiry date after the system configured number of days (90, in the example below):

Keywords	Number	Status	Expires	
voteb	50505	QUARANTINED	29-May-2013	 
votec	50505	RESERVED	N/A	

Showing 1 to 2 of 2 entries

- To reactivate the keyword, to RESERVED status, click the **green icon** on the right.
- To purge the keyword, click the **red cross icon** on the right.

D SMS Sandbox Service

The sandbox service replicates real U.S. Cellular Open API SMS web service and returns response objects, or 'canned responses', against pre-configured subscriber CTN values. It does not connect to any external interface. Developers can use this service to test different scenarios of their application without connecting to the real subscriber profile service.

Developers should use the SandboxDataService API to preconfigure responses for subscriber addresses. Details are described in a separate SandboxDataService API guide.

Code examples are provided below.

Send SMS Sandbox Request Example

```
POST
https://developerportal.uscellular.com/services/smsSendSandbox/2_0/smsmessaging/outbound/
12345/requests HTTP/1.1

Content-Type: application/json
Authorization: Basic QWVwb25hsdfdfgcDohUWF6eHN3MjNI
Content-Length: 323
Host: developerportal.uscellular.com
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)

{"outboundSMSMessageRequest":
  {
    "address": ["tel:+199999911111"],
    "clientCorrelator": "1045682",
    "outboundSMSTextMessage": {"message": "test"},
    "receiptRequest": {"notifyURL": "http://example.com/MockRestService/rest"},
    "senderAddress": "12345",
    "senderName": "Aepona"
  }
}
```

Query SMS Delivery Status Request Example

```
GET
```

```
https://developerportal.uscellular.com/services/smsSendSandbox/2_0/smsmessaging/outbound/  
12345/requests/66400037/deliveryInfos HTTP/1.1
```

```
Authorization: Basic QWVwb25hVsdffsfgDohUWF6eHN3MjNI
```

```
Host: developerportal.uscellular.com
```

```
Connection: Keep-Alive
```

```
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

End of Document