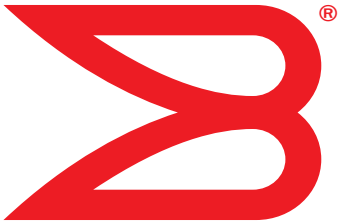
53-1003676-01
30 April, 2015



Brocade Flow Optimizer

REST API Guide

Supporting Flow Optimizer 1.0

BROCADE

Brocade Communications Systems, Incorporated

Corporate and Latin American Headquarters
Brocade Communications Systems, Inc.
130 Holger Way
San Jose, CA 95134
Tel: 1-408-333-8000
Fax: 1-408-333-8101
E-mail: info@brocade.com

Asia-Pacific Headquarters
Brocade Communications Systems China HK, Ltd.
No. 1 Guanghua Road
Chao Yang District
Units 2718 and 2818
Beijing 100020, China
Tel: +8610 6588 8888
Fax: +8610 6588 9999
E-mail: china-info@brocade.com

European Headquarters
Brocade Communications Switzerland Sàrl
Centre Swissair
Tour B - 4ème étage
29, Route de l'Aéroport
Case Postale 105
CH-1215 Genève 15
Switzerland
Tel: +41 22 799 5640
Fax: +41 22 799 5641
E-mail: emea-info@brocade.com

Asia-Pacific Headquarters
Brocade Communications Systems Co., Ltd. (Shenzhen WFOE)
Citic Plaza
No. 233 Tian He Road North
Unit 1308 – 13th Floor
Guangzhou, China
Tel: +8620 3891 2000
Fax: +8620 3891 2111
E-mail: china-info@brocade.com

Document History

Title	Publication number	Summary of changes	Date
Brocade Flow Optimizer REST API Guide 1.0	53-1003676-01	New document	April 2015

Contents

Getting Started

In this chapter

How this document is organized

This document is organized to help you find the information that you want as quickly and easily as possible.

The document contains the following components:

- Chapter 1, “REST API Namespace and Resources,” describes the REST API namespace “ns0” (URI and XSD) and resources.
- Chapter 2, “REST API Data Model,” describes the REST API data model including data elements and data types.
- Chapter 3, “REST API Files and Libraries,” describes the REST API files and libraries.
- Appendix A, “REST API Schemas,” lists the schemas used by REST API calls, including REST API files and libraries schemas, namespace “ns0” schema, and WADL.

Before you begin

This document assumes that you are familiar with the concept of REST APIs.

Before you can use the Flow Optimizer REST API:

- Make sure that Flow Optimizer 1.0 or later is installed on your network.
- Obtain a username and password for accessing Flow Optimizer through the REST API.
- Make sure that you have a tool for interacting with REST APIs.

Document conventions

This section describes text formatting conventions and important notice formats used in this document.

Text formatting

The narrative-text formatting conventions that are used are as follows:

bold text	Identifies command names Identifies the names of user-manipulated GUI elements Identifies keywords and operands Identifies text to enter at the GUI or CLI
italic text	Provides emphasis Identifies variables Identifies paths and Internet addresses Identifies document titles
`code` text	Identifies CLI output Identifies command syntax examples

Notes, cautions, and warnings

The following notices and statements are used in this manual. They are listed below in order of increasing severity of potential hazards.

NOTE
A note provides a tip, guidance, or advice, emphasizes important information, or provides a reference to related information.

ATTENTION
An Attention statement indicates potential damage to hardware or data.

CAUTION
A Caution statement alerts you to situations that can be potentially hazardous to you or cause damage to hardware, firmware, software, or data.

DANGER
A Danger statement indicates conditions or situations that can be potentially lethal or extremely hazardous to you. Safety labels are also attached directly to products to warn of these conditions or situations.

Key terms

For definitions specific to Brocade and Fibre Channel, see the technical glossaries on MyBrocade. Refer to "Brocade resources" on page ix for instructions on accessing MyBrocade.

For definitions of SAN-specific terms, visit the Storage Networking Industry Association online dictionary at:

http://www.snia.org/education/dictionary

Notice to the reader

This document may contain references to the trademarks of the following corporations. These trademarks are the properties of their respective companies and corporations.

These references are made for informational purposes only.

Corporation	Referenced Trademarks and Products
Microsoft Corporation	Windows, Windows NT, Internet Explorer
Oracle Corporation	Oracle, Java
Netscape Communications Corporation	Netscape
Red Hat, Inc.	Red Hat, Red Hat Network, Maximum RPM, Linux Undercover

Additional information

This section lists additional Brocade and industry-specific documentation that you might find helpful.

Brocade resources

To get up-to-the-minute information, go to http://my.brocade.com to register at no cost for a user ID and password.

White papers, online demonstrations, and data sheets are available through the Brocade website at:

http://www.brocade.com/products-solutions/products/index.page

For additional Brocade documentation, visit the Brocade website:

http://www.brocade.com

Release notes are available on the MyBrocade website.

Other industry resources

For additional resource information, visit the Technical Committee T11 website. This website provides interface standards for high-performance and mass storage applications for Fibre Channel, storage management, and other applications:

http://www.t11.org

For information about the Fibre Channel industry, visit the Fibre Channel Industry Association website:

http://www.fibrechannel.org

Getting technical help

Contact your switch support supplier for hardware, firmware, and software support, including product repairs and part ordering. To expedite your call, have the following information available:

1. General Information
    - Switch model
    - Switch operating system version
    - Software name and software version, if applicable
    - Error numbers and messages received
    - Detailed description of the problem, including the switch or fabric behavior immediately following the problem, and specific questions
    - Description of any troubleshooting steps already performed and the results
    - Serial console and Telnet session logs
    - syslog message logs
2. Switch Serial Number

    The switch serial number and corresponding bar code are provided on the serial number label, as illustrated below:

    The serial number label is located on the switch ID pull-out tab located on the bottom of the port side of the switch.
3. World Wide Name (WWN)

    Use the **show license id** command to display the WWN of the chassis.

    If you cannot use the **show license id** command because the switch is inoperable, you can get the WWN from the same place as the serial number, except for the Brocade DCX. For the Brocade DCX, access the numbers on the WWN cards by removing the Brocade logo plate at the top of the nonport side of the chassis.

Document feedback

Quality is our first concern at Brocade and we have made every effort to ensure the accuracy and completeness of this document. However, if you find an error or an omission, or you think that a topic needs further development, we want to hear from you. Forward your feedback to:

documentation@brocade.com

Provide the title and version number of the document and as much detail as possible about your comment, including the topic heading and page number and your suggestions for improvement.

Chapter

REST API Namespace and Resources

1

In this chapter

Namespace “ns0”

The Namespace URI and XSD are as follows:

Namespace URI:	(default namespace)
XSD:	ns0.xsd

NOTE
You must click on the filename for the XSD in column two to view the underlying XML code. If viewing this document in Acrobat, click ALT-LeftArrow to return to this page.

REST Resources

This API supports a Representational State Transfer (REST) model for accessing a set of resources through a fixed set of operations.

- The REST resources expose a data model that is supported by a set of client-side libraries that are made available in the /users/{username} topic.
- There is also a Namespace “ns0” XML Schema Examples topic providing REST API examples.

The following resources are accessible through the RESTful model.

EventsApi

The following resource is applicable: /events (Mount Point: /collector/events). The following operations are supported on this resource:

- Get
- Options

Get

Returns all events for the following:

Parameters

Name	Description	Type	Default
timeline	- The time from which the duration is referenced. This is a long in milliseconds.	query	
duration	- the duration from the timeline for the search. This is a long in milliseconds.	query	1800000
offset	- the numerical offset from 0 to get events in pages.	query	
limit	- the number of events per page.	query	

Response Body

element:	events
media types:	application/xml application/json

Events Object

Contains a list of events.

Example

```
Request URL:
https://localhost:8089/collector/events

Request Headers for XML output:
Authorization - Guest_1426270596791
Accept - Application/xml

Response body in XML:
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<events>
   <event>
      <description>Profile Added/Updated Successfully.</description>
      <category>AUDIT</category>
      <messageID>AUD-1005</messageID>
      <timestamp>1420842996921</timestamp>
      <severity>INFO</severity>
      <profileAction>DROP</profileAction>
      <profileName>NTP Reflection</profileName>
   </event>
</events>

Request Headers For JSON output:
Authorization - Administrator_1426270596791
Accept - Application/json

Response body in JSON:
{
"event":
[
{
"description": "Profile Added/Updated Successfully.",
"category": "Audit Event",
"messageID": "AUD-1005",
"timeStamp": 1420842996921,
```

```
"severity": "INFO",
"profileName": "NTP Reflection",
"profileAction": "NONE"
}
]
}
```

Status Codes

HTTP Status Code	Description
200 OK	SUCCESS
500 Internal Server Error	FAILED, reason included in HTTP error response.

Options

Response Body

element:	(custom)
media types:	application/xml application/json

Status Codes

HTTP Status Code	Description
200 OK	SUCCESS

LoginApi

The following resource is applicable: /login (Mount Point: /collector/login). The following operations are supported on this resource:

- Post
- Options

Post

Allows user to login to the app using a valid username and password.

Request Body

element:	login
media types:	application/xml application/json

NOTE
Login object contains valid username and password.

Response Body

element:	login
media types:	application/xml application/json

NOTE
Login object contains the token that can be used for authorization of other REST requests.

Example

```
Request URL:
https://localhost:8089/collector/login

Request Headers For XML:
Accept - Application/xml
Content-Type - Application/xml

Request payload in XML:
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<login>
   <username>Administrator</username>
   <password>pass</password>
</login>

Response body in XML:
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<login>
  <token>Administrator_1426270596791</token>
  <status>Login success</status>
</login>

Request Headers For JSON:
Accept - Application/json
Content-Type - Application/json

Request payload in JSON:
{
"username": "Administrator",
"password": "pass"
}

Response body in JSON:
{
"token": "Administrator_1426620456948",
"status": "Login success"
}
```

Status Codes

HTTP Status Code	Description
200 OK	SUCCESS
401 Unauthorized	FAILED, reason included in HTTP error response.

Options

Response Body

element:	(custom)
media types:	application/xml application/json

Status Codes

HTTP Status Code	Description
200 OK	SUCCESS

LogoutApi

The following resource is applicable: /logout (Mount Point: /collector/logout). The following operations are supported on this resource:

- Post
- Options

Post

Allows user to logout of the app.

Response Body

element:	login
media types:	application/xml application/json

NOTE
Allows user to logout of the app.

Example

```
Request URL:
https://localhost:8089/collector/logout

Request Headers For XML:
Authorization - Administrator_1426270596791
Accept - Application/xml

Response body in XML:
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<login>
  <status>Session Administrator_1426270596791 removed</status>
```

```
</login>

Request Headers For JSON:
Authorization - Administrator_1426270596791
Accept - Application/json

Response body in JSON:
{
"status": "Session Administrator_1426270596791 removed"
}
```

Status Codes

HTTP Status Code	Description
200 OK	SUCCESS
401 Unauthorized	FAILED, reason included in HTTP error response.

Options

Response Body

element:	(custom)
media types:	application/xml application/json

Status Codes

HTTP Status Code	Description
200 OK	SUCCESS

ProfilesApi

The following resource is applicable: /profiles (Mount Point: /collector/profiles). The following operations are supported on this resource:

- Get
- Put
- Post
- Delete
- Options

Get

Returns a list of Profiles configured in VTM App.

Response Body

element:	profiles
media types:	application/xml application/json

NOTE
List of profiles configured in VTM App.

Example

```
Request URL:
https://localhost:8089/collector/profiles

Request Headers for XML output:
Accept - Application/xml
Authorization - Guest_1426270596791

Response body in XML:
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<profiles>
   <profile>
      <id>1</id>
      <name>NTP Reflection</name>
      <network_attribuites>
         <network_attribuite>
            <name>protocol</name>
            <value>UDP</value>
         </network_attribuite>
         <network_attribuite>
            <name>source-port</name>
            <value>123</value>
         </network_attribuite>
      </network_attribuites>
      <redirect_nodes>
         <redirect_node>
            <id>00:24:38:80:8e:00</id>
            <ports>49,50,51</ports>
         </redirect_node>
         <redirect_node>
            <id>00:24:38:80:5r:00</id>
            <ports>123</ports>
         </redirect_node>
      </redirect_nodes>
      <observation_time>15000</observation_time>
      <threshold>5000</threshold>
      <reduced_threshold>4500</reduced_threshold>
      <action>DROP</action>
      <last_modified_time>1418261199762</last_modified_time>
      <last_modified_by>Administrator</last_modified_by>
      <priority>1</priority>
      <enabled>0</enabled>
      <type>0</type>
      <rate_limit>3000</rate_limit>
   </profile>
</profiles>

Request Headers for JSON output:
```

```
Accept - Application/json
Authorization - Guest_1426270596791

Response body in JSON:
{
"profile":
[
{
"id": 1,
"name": "NTP Reflection",
"threshold": 5000,
"action": "DROP",
"enabled": 0,
"type": 0,
"priority": 1,
"observation_time": 15000,
"reduced_threshold": 4500,
"last_modified_time": 1418261199762,
"last_modified_by": "Administrator",
"rate_limit": 3,
"network_attributes":
{
"network_attribute":
[
{
"name": "protocol",
"value": "UDP"
},
{
"name": "source-port",
"value": "123"
}
]
},
"redirect_nodes":
{
"redirect_node":
[
{
"id": "00:24:38:80:8e:00",
"ports": "5"
},
{
"id": "00:24:38:80:5r:00",
"ports": "2,3"
}
]
}
}
]
}
```

Status Codes

HTTP Status Code	Description
200 OK	Operation successful.

Put

Modify a Profile in VTM Application. If the Profile already exists, it will update the current Profile.

Parameters

Name	Description	Type	Default
profileName	Name of the Profile	query	

Request Body

element:	profile
media types:	application/xml application/json

Response Body

element:	(custom)
media types:	*/* application/xml

NOTE
Response as dictated by the HTTP Response Status code.

Example

```
Request URL:
https://localhost:8089/collector/profiles

Request Headers for XML output:
Accept - Application/xml
Authorization - Guest_1426270596791

Request body in XML:
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<profile>
   <id>1</id>
   <name>NTP Reflection</name>
   <network_attribuites>
      <network_attribuite>
         <name>protocol</name>
         <value>UDP</value>
      </network_attribuite>
      <network_attribuite>
         <name>source-port</name>
         <value>123</value>
      </network_attribuite>
   </network_attribuites>
   <redirect_nodes>
      <redirect_node>
         <id>00:24:38:80:8e:00</id>
```

```
            <ports>49,50,51</ports>
         </redirect_node>
         <redirect_node>
            <id>00:24:38:80:5r:00</id>
            <ports>123</ports>
         </redirect_node>
      </redirect_nodes>
      <observation_time>15000</observation_time>
      <threshold>5000</threshold>
      <reduced_threshold>4500</reduced_threshold>
      <action>DROP</action>
      <last_modified_time>1418261199762</last_modified_time>
      <last_modified_by>Administrator</last_modified_by>
      <priority>1</priority>
      <enabled>0</enabled>
      <type>0</type>
      <rate_limit>3000</rate_limit>
</profile>

Request Headers for JSON output:
Accept - Application/json
Authorization - Guest_1426270596791

Request body in JSON:

{
"id": 1,
"name": "NTP Reflection",
"threshold": 5000,
"action": "DROP",
"enabled": 0,
"type": 0,
"priority": 1,
"observation_time": 15000,
"reduced_threshold": 4500,
"last_modified_time": 1418261199762,
"last_modified_by": "Administrator",
"rate_limit": 3,
"network_attributes":
{
"network_attribute":
[
{
"name": "protocol",
"value": "UDP"
},
{
"name": "source-port",
"value": "123"
}
]
},
"redirect_nodes":
{
"redirect_node":
[
{
"id": "00:24:38:80:8e:00",
"ports": "5"
},
```

```
{
"id": "00:24:38:80:5r:00",
"ports": "2,3"
}
]
}
}
```

Status Codes

HTTP Status Code	Description
204 No Content	Profile Updated successfully
500 Internal Server Error	Failed to update Profile. Failure Reason included in HTTP Error response.

Post

Response as dictated by the HTTP Response Status code.

Request Body

element:	profile
media types:	application/xml application/json

Response Body

element:	(custom)
media types:	*/* application/xml

NOTE
Response as dictated by the HTTP Response Status code.

Example

```
Request URL:
https://localhost:8089/collector/profiles

Request Headers for XML output:
Accept - Application/xml
Authorization - Guest_1426270596791

Request body in XML:
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<profile>
   <id>1</id>
   <name>NTP Reflection</name>
   <network_attribuites>
      <network_attribuite>
         <name>protocol</name>
         <value>UDP</value>
      </network_attribuite>
      <network_attribuite>
         <name>source-port</name>
         <value>123</value>
```

```
      </network_attribuite>
   </network_attributes>
   <redirect_nodes>
      <redirect_node>
         <id>00:24:38:80:8e:00</id>
         <ports>49,50,51</ports>
      </redirect_node>
      <redirect_node>
         <id>00:24:38:80:5r:00</id>
         <ports>123</ports>
      </redirect_node>
   </redirect_nodes>
   <observation_time>15000</observation_time>
   <threshold>5000</threshold>
   <reduced_threshold>4500</reduced_threshold>
   <action>DROP</action>
   <last_modified_time>1418261199762</last_modified_time>
   <last_modified_by>Administrator</last_modified_by>
   <priority>1</priority>
   <enabled>0</enabled>
   <type>0</type>
   <rate_limit>3000</rate_limit>
</profile>

Request Headers for JSON output:
Accept - Application/json
Authorization - Guest_1426270596791

Request body in JSON:

{
"id": 1,
"name": "NTP Reflection",
"threshold": 5000,
"action": "DROP",
"enabled": 0,
"type": 0,
"priority": 1,
"observation_time": 15000,
"reduced_threshold": 4500,
"last_modified_time": 1418261199762,
"last_modified_by": "Administrator",
"rate_limit": 3,
"network_attributes":
{
"network_attribute":
[
{
"name": "protocol",
"value": "UDP"
},
{
"name": "source-port",
"value": "123"
}
]
},
"redirect_nodes":
{
"redirect_node":
```

```
[
{
"id": "00:24:38:80:8e:00",
"ports": "5"
},
{
"id": "00:24:38:80:5r:00",
"ports": "2,3"
}
]
}
}
```

Status Codes

HTTP Status Code	Description
201 Created	Profile added successfully.
500 Internal Server Error	Failed to add Profile. Failure Reason included in HTTP Error response.

Delete

Delete Profile in VTM Application.

Parameters

Name	Description	Type	Default
profileName	Name of the Profile to be deleted	query	

Response Body

element:	(custom)
media types:	*/* application/xml

NOTE
Response as dictated by the HTTP Response Status code.

Example

```
Request Headers for XML output:
Accept - Application/xml
Authorization - Guest_1426270596791
```

Example

```
Request URL:
https://localhost:8089/collector/profiles

Request body in XML:
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<profile>
   <id>1</id>
   <name>NTP Reflection</name>
   <network_attribuites>
```

```
      <network_attribuite>
         <name>protocol</name>
         <value>UDP</value>
      </network_attribuite>
      <network_attribuite>
         <name>source-port</name>
         <value>123</value>
      </network_attribuite>
   </network_attribuites>
   <redirect_nodes>
      <redirect_node>
         <id>00:24:38:80:8e:00</id>
         <ports>49,50,51</ports>
      </redirect_node>
      <redirect_node>
         <id>00:24:38:80:5r:00</id>
         <ports>123</ports>
      </redirect_node>
   </redirect_nodes>
   <observation_time>15000</observation_time>
   <threshold>5000</threshold>
   <reduced_threshold>4500</reduced_threshold>
   <action>DROP</action>
   <last_modified_time>1418261199762</last_modified_time>
   <last_modified_by>Administrator</last_modified_by>
   <priority>1</priority>
   <enabled>0</enabled>
   <type>0</type>
   <rate_limit>3000</rate_limit>
</profile>

Request Headers for JSON output:
Accept - Application/json
Authorization - Guest_1426270596791

Request body in JSON:

{
"id": 1,
"name": "NTP Reflection",
"threshold": 5000,
"action": "DROP",
"enabled": 0,
"type": 0,
"priority": 1,
"observation_time": 15000,
"reduced_threshold": 4500,
"last_modified_time": 1418261199762,
"last_modified_by": "Administrator",
"rate_limit": 3,
"network_attributes":
{
"network_attribute":
[
{
"name": "protocol",
"value": "UDP"
},
{
"name": "source-port",
```

```
"value": "123"
}
]
},
"redirect_nodes":
{
"redirect_node":
[
{
"id": "00:24:38:80:8e:00",
"ports": "5"
},
{
"id": "00:24:38:80:5r:00",
"ports": "2,3"
}
]
}
}
```


Status Codes

HTTP Status Code	Description
204 No Content	Profile deleted successfully
500 Internal Server Error	Failed to delete Profile. Failure Reason included in HTTP Error response.

Options

Response Body

element:	profiles
media types:	application/xml application/json

Status Codes

HTTP Status Code	Description
200 OK	SUCCESS

UserApi

The follows resources are applicable for the UserApi:

- /users
- /users/{username}

/users

The following resource is applicable: /users (Mount Point: /collector/users). The following operations are supported on this resource:

- Get
- Post

- Options

Get

Returns all users. Any user can perform this operation.

Response Body

element:	users
media types:	application/xml application/json

NOTE
Contains a list of Users.

Example

```
Request URL:
https://localhost:8089/collector/users

Request Headers for XML output:
Authorization - Guest_1426270596791
Accept - Application/xml

Response body in XML:
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<users>
   <user>
      <name>Administrator</name>
      <access_privilege>-1</access_privilege>
      <created_on>Tue Dec 23 09:24:50 PST 2014</created_on>
      <last_modified_on>Tue Dec 23 09:24:50 PST 2014</last_modified_on>
   </user>
   <user>
      <name>Guest</name>
      <access_privilege>0</access_privilege>
      <created_on>Fri Mar 13 10:31:33 PDT 2015</created_on>
      <last_modified_on>Fri Mar 13 10:31:33 PDT 2015</last_modified_on>
   </user>
</users>

Request Headers For JSON output:
Authorization - Administrator_1426270596791
Accept - Application/json

Response body in JSON:
{
"user": [
{
"name": "Administrator",
"access_privilege": -1,
"created_on": "Tue Dec 23 09:24:50 PST 2014",
"last_modified_on": "Tue Dec 23 09:24:50 PST 2014"
},
{
"name": "Guest",
"access_privilege": 0,
```

```
"created_on": "Fri Mar 13 10:31:33 PDT 2015",
"last_modified_on": "Fri Mar 13 10:31:33 PDT 2015"
}
]
}
```

Status Codes

HTTP Status Code	Description
200 OK	SUCCESS
500 Internal Server Error	FAILED, reason included in HTTP error response.

Post

Creates single new user. Only user with Admin privileges (**access_privilege=1**) is allowed to create a new user. Valid name and password are required as input. Access privilege is optional, if not provided, created user will have operational privileges (**access_privilege=0**). One cannot create user with same name.

Request Body

element:	schemaUser
media types:	application/xml application/json

Response Body

element:	(custom)
media types:	*/* application/xml

Example

```
Request URL:
https://localhost:8089/collector/users

Request Headers For XML input:
Authorization - Administrator_1426270596791
Content-Type - Application/xml

Request payload in XML:
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<user>
   <name>GuestOne</name>
   <password>guest_one</password>
   <access_privilege>0</access_privilege>
</user>

Request Headers For JSON input:
Authorization - Administrator_1426270596791
Content-Type - Application/json

Request payload in JSON:
{
"name": "GuestTwo",
"password": "guest_two",
```

```
"access_privilege": 1
}
```

Status Codes

HTTP Status Code	Description
201 Created	SUCCESS
500 Internal Server Error	FAILED, reason included in HTTP error response.

Options

Response Body

element:	(custom)
media types:	application/xml application/json

Status Codes

200 OK	SUCCESS

/users/{username}

The following resource is applicable: /users (Mount Point: /collector/users/{username}). The following operations are supported on this resource:

- Put
- Delete

Put

Updates single user. Only user with Admin privileges (**access_privilege=1**) is allowed to update an existing user. Password and **access_privilege** can be updated. User by name Administrator is root user and cannot be updated.

Parameters

Name	Description	Type	Default
username	The path parameter, which is the name of the user to be updated.	path	

Request Body

element:	schemaUser
media types:	application/xml application/json

NOTE
The User object containing properties to be updated.

Response Body

element:	(custom)
media types:	*/* application/xml

Example

```
Request URL:
https://localhost:8089/collector/users/GuestOne

Request Headers For XML input:
Authorization - Administrator_1426270596791
Content-Type - Application/xml

Request payload in XML:
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<user>
   <password>guest_one_updated</password>
</user>

Request URL:
https://localhost:8089/collector/users/GuestTwo

Request Headers For JSON input:
Authorization - Administrator_1426270596791
Content-Type - Application/json

Request payload in JSON:
{
"password": "guest_two_updated",
"access_privilege": 0
}
```

Status Codes

HTTP Status Code	Description
204 No Content	SUCCESS
500 Internal Server Error	FAILED, reason included in HTTP error response.

Delete

Deletes single user. Only user with Admin privileges (**access_privilege=1**) is allowed to delete an existing user. User by name Administrator is root user and cannot be deleted.

Parameters

Name	Description	Type	Default
username	The path parameter, which is the name of the user to be deleted.	path	

Response Body

element:	(custom)
media types:	*/* application/xml

Example

```
Request URL:
https://localhost:8089/collector/users/GuestOne

Request Headers:
Authorization - Administrator_1426270596791
```

Status Codes

HTTP Status Code	Description
204 No Content	SUCCESS
500 Internal Server Error	FAILED, reason included in HTTP error response.

Chapter

REST API Data Model

2

In this chapter

Data Model

The REST API data model includes the following topics:

- Data Elements
- Data Types

Data Elements

Each data element includes the a schema fragment specifying the expected content contained within that class, example XML code, and example JSON code.

events

Type

Namespace:	(default namespace)
XML Schema:	Namespace “ns0”

Example XML

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<events>
  <timeline>...</timeline>
  <duration>...</duration>
  <offset>...</offset>
  <limit>...</limit>
  <event>
    <description>...</description>
    <category>...</category>
    <messageID>...</messageID>
    <timestamp>...</timestamp>
    <severity>...</severity>
    <profileName>...</profileName>
    <profileAction>...</profileAction>
  </event>
  <event>
    <!--...-->
  </event>
  <!--...more "event" elements...-->
</events>
```

Example JSON

```
{
  "timeline" : ...,
  "duration" : ...,
  "offset" : ...,
  "limit" : ...,
  "event" : [ {
    "description" : "...",
    "category" : "...",
    "messageID" : "...",
    "timestamp" : ...,
    "severity" : "...",
    "profileName" : "...",
    "profileAction" : "..."
  }, ... ]
}
```

login

Type

Namespace:	(default namespace)
XML Schema:	Namespace “ns0”

The following schema fragment specifies the expected content contained within this class.

```
<complexType>
<complexContent>
<restriction base="{http://www.w3.org/2001/XMLSchema}anyType">
<sequence>
<element name="username" type="{http://www.w3.org/2001/XMLSchema}string"/>
<element name="password" type="{http://www.w3.org/2001/XMLSchema}string"/>
<element name="token" type="{http://www.w3.org/2001/XMLSchema}string"/>
<element name="status" type="{http://www.w3.org/2001/XMLSchema}string"/>
</sequence>
</restriction>
</complexContent>
</complexType>
```

Example XML

```
<?xml version="1.0" encoding="UTF-8"?>
<login>
  <username>...</username>
  <password>...</password>
  <token>...</token>
  <status>...</status>
</login>
```

Example JSON

```
{
  "username" : "...",
  "password" : "...",
  "token" : "...",
  "status" : "..."
}
```

profiles

Type

Namespace:	(default namespace)
XML Schema:	Namespace “ns0”

The following schema fragment specifies the expected content contained within this class.

```
<complexType>
<complexContent>
<restriction base="{http://www.w3.org/2001/XMLSchema}anyType">
<sequence>
<element name="profile" type="{}profile" maxOccurs="unbounded"
minOccurs="0"/>
</sequence>
</restriction>
</complexContent>
</complexType>
```

Example XML

```
<?xml version="1.0" encoding="UTF-8"?>
<profiles>
  <profile>
    <id>...</id>
    <name>...</name>
    <description>...</description>
    <observation_interval_ms>...</observation_interval_ms>
    <threshold>...</threshold>
    <reduced_threshold>...</reduced_threshold>
    <action>...</action>
    <last_modified_time>...</last_modified_time>
    <last_modified_by>...</last_modified_by>
    <enabled>...</enabled>
    <type>...</type>
    <priority>...</priority>
    <rate_limit>...</rate_limit>
    <dscp_rate_limit>...</dscp_rate_limit>
    <dscp_prec_level>...</dscp_prec_level>
```

```
    <flow_timeout>...</flow_timeout>
    <ingress_ports>
      <redirect_node>
        <id>...</id>
        <ports>...</ports>
      </redirect_node>
      <redirect_node>
        <!--...-->
      </redirect_node>
      <!--...more "redirect_node" elements...-->
    </ingress_ports>
    <network_attributes>
      <network_attribute>
        <name>...</name>
        <value>...</value>
        <label>...</label>
      </network_attribute>
      <network_attribute>
        <!--...-->
      </network_attribute>
      <!--...more "network_attribute" elements...-->
    </network_attributes>
    <redirect_nodes>
      <redirect_node>
        <id>...</id>
        <ports>...</ports>
      </redirect_node>
      <redirect_node>
        <!--...-->
      </redirect_node>
      <!--...more "redirect_node" elements...-->
    </redirect_nodes>
  </profile>
  <profile>
    <!--...-->
  </profile>
  <!--...more "profile" elements...-->
</profiles>
```

Example JSON

```
{
  "profile" : [ {
    "id" : ...,
    "name" : "...",
    "description" : "...",
    "observation_interval_ms" : ...,
    "threshold" : ...,
    "reduced_threshold" : ...,
    "action" : "...",
    "last_modified_time" : ...,
    "last_modified_by" : "...",
    "enabled" : ...,
    "type" : ...,
    "priority" : ...,
    "rate_limit" : ...,
    "dscp_rate_limit" : ...,
    "dscp_prec_level" : ...,
    "flow_timeout" : ...,
    "ingress_ports" : {
```

```
      "redirect_node" : [ {
        "id" : "...",
        "ports" : "..."
      }, ... ]
    },
    "network_attributes" : {
      "network_attribute" : [ {
        "name" : "...",
        "value" : "...",
        "label" : "..."
      }, ... ]
    },
    "redirect_nodes" : {
      "redirect_node" : [ {
        "id" : "...",
        "ports" : "..."
      }, ... ]
    }
  }, ... ]
}
```

users

Type

Namespace:	(default namespace)
XML Schema:	Namespace “ns0”

The following schema fragment specifies the expected content contained within this class.

```
<complexType>
<complexContent>
<restriction base="{http://www.w3.org/2001/XMLSchema}anyType">
<sequence>
<element name="user" type="{}user" maxOccurs="unbounded"/>
</sequence>
</restriction>
</complexContent>
</complexType>
```

Example XML

```
<?xml version="1.0" encoding="UTF-8"?>
<users>
  <user>
    <name>...</name>
    <password>...</password>
    <access_privilege>...</access_privilege>
    <created_on>...</created_on>
    <last_modified_on>...</last_modified_on>
  </user>
  <user>
    <!--...-->
  </user>
  <!--...more "user" elements...-->
</users>
```

Example JSON

```
{
  "user" : [ {
    "name" : "...",
    "password" : "...",
    "access_privilege" : ...,
    "created_on" : "...",
    "last_modified_on" : "..."
  }, ... ]
}
```

Data Types

Each data type includes the a schema fragment specifying the expected content contained within that class, example XML code, and example JSON code.

event

The following XML Elements and JSON tables specify the XML and JSON elements contained within this class.

XML Elements

The following XML elements table contains the name (type), minimum/maximum occurs, and description for this data type:

Name (type)	Min/Max Occurs	Description
description (string)	1/1	
category (string)	1/1	
messageID (string)	1/1	
timestamp (long)	1/1	
severity (string)	1/1	
profileName (string)	0/1	
profileAction (string)	1/1	

JSON

The following JSON table contains the property, type, and description for this data type:

Property	Type	Description
description	description (string)	
category	category (string)	
messageID	messageID (string)	
timestamp	timestamp (long)	
severity	severity (string)	
profileName	profileName (string)	
profileAction	profileAction (string)	

network_attribute

The following schema fragment specifies the expected content contained within this class.

```
<complexType name="network_attribute">
<complexContent>
<restriction base="{http://www.w3.org/2001/XMLSchema}anyType">
<sequence>
<element name="name" type="{http://www.w3.org/2001/XMLSchema}string"/>
<element name="value" type="{http://www.w3.org/2001/XMLSchema}string"/>
<element name="label" type="{http://www.w3.org/2001/XMLSchema}string"/>
</sequence>
</restriction>
</complexContent>
</complexType>
```

The following XML Elements and JSON tables specify the XML and JSON elements contained within this class.

XML Elements

The following XML elements table contains the name (type), minimum/maximum occurs, and description for this data type:

Name (type)	Min/Max Occurs	Description
name (string)	1/1	
value (string)	1/1	
label (string)	1/1	

JSON

The following JSON table contains the property, type, and description for this data type:

Property	Type	Description
name	name (string)	
value	value (string)	
label	label (string)	

network_attributes

The following schema fragment specifies the expected content contained within this class.

```
<complexType name="network_attributes">
<complexContent>
<restriction base="{http://www.w3.org/2001/XMLSchema}anyType">
<sequence>
<element name="network_attribute" type="{}network_attribute"
maxOccurs="unbounded" minOccurs="0"/>
</sequence>
</restriction>
</complexContent>
</complexType>
```

The following XML Elements and JSON tables specify the XML and JSON elements contained within this class.

XML Elements

The following XML elements table contains the name (type), minimum/maximum occurs, and description for this data type:

Name (type)	Min/Max Occurs	Description
network_attribute (network_attribute)	0/unbounded	

JSON

The following JSON table contains the property, type, and description for this data type:

Property	Type	Description
network_attribute	array of network_attribute (network_attribute)	

profile

The following schema fragment specifies the expected content contained within this class.

```
<complexType name="profile">
<complexContent>
<restriction base="{http://www.w3.org/2001/XMLSchema}anyType">
<sequence>
<element name="id" type="{http://www.w3.org/2001/XMLSchema}int"
minOccurs="0"/>
<element name="name" type="{http://www.w3.org/2001/XMLSchema}string"
minOccurs="0"/>
<element name="description" type="{http://www.w3.org/2001/XMLSchema}string"
minOccurs="0"/>
<element name="observation_interval_ms"
type="{http://www.w3.org/2001/XMLSchema}int" minOccurs="0"/>
<element name="threshold" type="{http://www.w3.org/2001/XMLSchema}int"
minOccurs="0"/>
<element name="reduced_threshold"
type="{http://www.w3.org/2001/XMLSchema}float" minOccurs="0"/>
<element name="action" type="{http://www.w3.org/2001/XMLSchema}string"
minOccurs="0"/>
<element name="last_modified_time"
type="{http://www.w3.org/2001/XMLSchema}long" minOccurs="0"/>
<element name="last_modified_by"
type="{http://www.w3.org/2001/XMLSchema}string" minOccurs="0"/>
<element name="enabled" type="{http://www.w3.org/2001/XMLSchema}int"
minOccurs="0"/>
<element name="type" type="{http://www.w3.org/2001/XMLSchema}int"
minOccurs="0"/>
<element name="priority" type="{http://www.w3.org/2001/XMLSchema}int"
minOccurs="0"/>
<element name="rate_limit" type="{http://www.w3.org/2001/XMLSchema}int"
minOccurs="0"/>
<element name="dscp_rate_limit" type="{http://www.w3.org/2001/XMLSchema}int"
minOccurs="0"/>
<element name="dscp_prec_level" type="{http://www.w3.org/2001/XMLSchema}int"
minOccurs="0"/>
<element name="flow_timeout" type="{http://www.w3.org/2001/XMLSchema}int"
minOccurs="0"/>
<element name="ingress_ports" type="{}redirect_nodes" minOccurs="0"/>
```

```
<element name="network_attributes" type="{}network_attributes"
minOccurs="0"/>
<element name="redirect_nodes" type="{}redirect_nodes" minOccurs="0"/>
</sequence>
</restriction>
</complexContent>
</complexType>
```

The following XML Elements and JSON tables specify the XML and JSON elements contained within this class.

XML Elements

The following XML elements table contains the name (type), minimum/maximum occurs, and description for this data type:

Name (type)	Min/Max Occurs	Description
id (int)	0/1	
name (string)	0/1	
description (string)	0/1	
observation_interval_ms (int)	0/1	
threshold (int)	0/1	
reduced_threshold (float)	0/1	
action (string)	0/1	
last_modified_time (long)	0/1	
last_modified_by (string)	0/1	
enabled (int)	0/1	
type (int)	0/1	
priority (int)	0/1	
rate_limit (int)	0/1	
dscp_rate_limit (int)	0/1	
dscp_prec_level (int)	0/1	
flow_timeout (int)	0/1	
ingress_ports (redirect_nodes)	0/1	
network_attributes (network_attributes)	0/1	
redirect_nodes (redirect_nodes)	0/1	

JSON

The following JSON table contains the property, type, and description for this data type:

Property	Type	Description
id	id (int)	
name	name (string)	
description	description (string)	
observation_interval_ms	observation_interval_ms (int)	

threshold	threshold (int)	
reduced_threshold	reduced_threshold (float)	
action	action (string)	
last_modified_time	last_modified_time (long)	
last_modified_by	last_modified_by (string)	
enabled	enabled (int)	
type	type (int)	
priority	priority (int)	
rate_limit	rate_limit (int)	
dscp_rate_limit	dscp_rate_limit (int)	
dscp_prec_level	dscp_prec_level (int)	
flow_timeout	flow_timeout (int)	
ingress_ports	ingress_ports (redirect_nodes)	
network_attributes	network_attributes (network_attributes)	
redirect_nodes	redirect_nodes (redirect_nodes)	

redirect_node

The following schema fragment specifies the expected content contained within this class.

```
<complexType name="redirect_node">
<complexContent>
<restriction base="{http://www.w3.org/2001/XMLSchema}anyType">
<sequence>
<element name="id" type="{http://www.w3.org/2001/XMLSchema}string"/>
<element name="ports" type="{http://www.w3.org/2001/XMLSchema}string"/>
</sequence>
</restriction>
</complexContent>
</complexType>
```

The following XML Elements and JSON tables specify the XML and JSON elements contained within this class.

XML Elements

The following XML elements table contains the name (type), minimum/maximum occurs, and description for this data type:

Name (type)	Min/Max Occurs	Description
id (string)	1/1	
ports (string)	1/1	

JSON

The following JSON table contains the property, type, and description for this data type:

Property	Type	Description
id	id (string)	
ports	ports (string)	

redirect_nodes

The following schema fragment specifies the expected content contained within this class.

```
<complexType name="redirect_nodes">
<complexContent>
<restriction base="{http://www.w3.org/2001/XMLSchema}anyType">
<sequence>
<element name="redirect_node" type="{}redirect_node" maxOccurs="unbounded"
minOccurs="0"/>
</sequence>
</restriction>
</complexContent>
</complexType>
```

The following XML Elements and JSON tables specify the XML and JSON elements contained within this class.

XML Elements

The following XML elements table contains the name (type), minimum/maximum occurs, and description for this data type:

Name (type)	Min/Max Occurs	Description
redirect_node (redirect_node)	0/unbounded	

JSON

The following JSON table contains the property, type, and description for this data type:

Property	Type	Description
redirect_node	array of redirect_node (redirect_node)	

user

The following schema fragment specifies the expected content contained within this class.

```
<complexType name="user">
<complexContent>
<restriction base="{http://www.w3.org/2001/XMLSchema}anyType">
<sequence>
<element name="name" type="{http://www.w3.org/2001/XMLSchema}string"/>
<element name="password" type="{http://www.w3.org/2001/XMLSchema}string"/>
<element name="access_privilege"
type="{http://www.w3.org/2001/XMLSchema}int"/>
<element name="created_on" type="{http://www.w3.org/2001/XMLSchema}string"
minOccurs="0"/>
<element name="last_modified_on"
type="{http://www.w3.org/2001/XMLSchema}string" minOccurs="0"/>
```

```
</sequence>
</restriction>
</complexContent>
</complexType>
```

The following XML Elements and JSON tables specify the XML and JSON elements contained within this class.

XML Elements

The following XML elements table contains the name (type), minimum/maximum occurs, and description for this data type:

Name (type)	Min/Max Occurs	Description
name (string)	1/1	
password (string)	1/1	
access_privilege (int)	1/1	
created_on (string)	0/1	
last_modified_on (string)	0/1	

JSON

The following JSON table contains the property, type, and description for this data type:

Property	Type	Description
name	name (string)	
password	password (string)	
access_privilege	access_privilege (int)	
created_on	created_on (string)	
last_modified_on	last_modified_on (string)	

Chapter

REST API Files and Libraries

3

In this chapter

Files and Libraries

In this Topic

C Client Library

The C module generates the source code for the ANSI-C-compatible data structures and (de)serialization functions that can be used in conjunction with the following XML C parser and toolkit:

Name	Source
libxml	http://xmlsoft.org/

The generated C source code depends on the **<time.h>**, **<string.h>**, and **<stdlib.h>** C standard libraries and the following APIs:

Name	Source

XML Reader API	http://xmlsoft.org/html/libxml-xmlreader.html
XML Writer API	http://xmlsoft.org/html/libxml-xmlwriter.html

REST XML Example

```
#include <full.c>
//...

xmlTextWriterPtr writer = ...; //set up the writer to the url.
full_ns0_anonymous_login *request_element = ...;
xmlTextReaderPtr reader = ...; //set up the reader to the url.
full_ns0_anonymous_login *response_element = ...;
//set up the full_ns0_anonymous_login...
xml_write_full_ns0_anonymous_login(writer, request_element);
response_element = xml_read_full_ns0_anonymous_login(reader);

//handle the response as needed...

//free the full_ns0_anonymous_login
free_full_ns0_anonymous_login(request_element);
//free the full_ns0_anonymous_login
free_full_ns0_anonymous_login(response_element);
```

Files

The following files are applicable to this library:

Name	Size	Description
full.c	387.11K	
enunciate-common.c	39.67K	Common code needed for all projects.

.NET Client Library

The .NET client-side library defines the classes that can be (de)serialized to/from XML. This is useful for accessing the REST endpoints that are published by this application.

REST XML Example

```
//read a resource from a REST url
Uri uri = new Uri(...);

XmlSerializer s = new XmlSerializer(
  typeof( Login )
);

  //Create the request object
WebRequest req = WebRequest.Create(uri);
WebResponse resp = req.GetResponse();
Stream stream = resp.GetResponseStream();
TextReader r = new StreamReader( stream );

Login order = (Login) s.Deserialize( r );

//handle the result as needed...
```

Files

The following files are applicable to this library:

Name	Size	Description
full-dotnet.zip	6.77K	Contains full.cs file.
full.cs	122K	Contained in full-dotnet.zip file.

Java Client Library

The Java client-side library is used to access the Web service API for this application.

The JAX-WS client-side library is used to provide the set of Java objects that can be serialized to/from XML using JAXB. This is useful for accessing the REST endpoints that are published by this application.

Name	Source
JAXB	https://jaxb.java.net/

REST XML Example (Raw JAXB)

```
java.net.URL url = new java.net.URL(baseURL + "/login");
JAXBContext context = JAXBContext.newInstance( Login.class, Login.class );
java.net.URLConnection connection = url.openConnection();
connection.setDoOutput(true);
connection.connect();

Unmarshaller unmarshaller = context.createUnmarshaller();
Marshaller marshaller = context.createMarshaller();
marshaller.marshal(login, connection.getOutputStream());
Login result = (Login) unmarshaller.unmarshal( connection.getInputStream() );
//handle the result as needed...
```

REST XML Example (Jersey Client)

```
com.sun.jersey.api.client.Client client =
com.sun.jersey.api.client.Client.create();

Login result = client.resource(baseUrl + "/login")
  .entity(login)
  .post(Login.class);

//handle the result as needed...
```

Files

The following files are applicable to this library:

Name	Size	Description

full-client.jar	32.00K	The binaries for the Java client library.
full-client-sources.jar	25.13K	The sources for the Java client library.

Java JSON Client Library

The Java client-side library is used to provide the set of Java objects that can be serialized to/from JSON using Jackson. This is useful for accessing the JSON REST endpoints that are published by this application.

REST XML Example

```
java.net.URL url = new java.net.URL(baseURL + "/login");
ObjectMapper mapper = new ObjectMapper();
java.net.URLConnection connection = url.openConnection();
connection.setDoOutput(true);
connection.connect();

mapper.writeValue(connection.getOutputStream(), login);
Login result = (Login) mapper.readValue( connection.getInputStream(),
Login.class );
//handle the result as needed...
```

Files

The following files are applicable to this library:

Name	Size	Description
full-json-client.jar	23.96K	The binaries for the Java JSON client library.
full-json-client-sources.jar	23.33K	The sources for the Java JSON client library.

Objective C Client Library

The Objective C module generates the source code for the Objective C classes and (de)serialization functions that can be used in conjunction with libxml2 to (de)serialize the REST resources as they are represented as XML data.

The generated Objective C source code depends on the XML Reader API and the XML Writer API as well as the base OpenStep foundation classes.

See C Client Library for further information on libxml2, the XML Reader API, and the XML Writer API.

REST XML Example

```
#import <full.h>
//...

FULLNS0Login *requestElement = [[FULLNS0Login alloc] init];
NSData *requestData; //data holding the XML for the request.
FULLNS0Login *responseElement;
NSData *responseData; //data holding the XML from the response.
```

```
NSURL *baseURL = ...; //the base url including the host and subpath.
NSURL *url = [NSURL URLWithString: @"/login" relativeToURL: baseURL];
NSMutableURLRequest *request = [[NSMutableURLRequest alloc] initWithURL:url];
NSURLResponse *response = nil;
NSError *error = NULL;
[request setHTTPMethod: @"POST"];
[request setValue:@"application/xml" forHTTPHeaderField:@"Content-Type"];

//set up the FULLNS0Login...

requestData = [requestElement writeToXML];
[request setHTTPBody: requestData];
```

Example

```
//this example uses a synchronous request,
//but you'll probably want to use an asynchronous call
responseData = [NSURLConnection sendSynchronousRequest:request
returningResponse:&response error:&error];
FULLNS0Login *responseElement = [FULLNS0Login readFromXML: responseData];
[responseElement retain];

//handle the response as needed...
```

Files

The following files are applicable to this library:

Name	Size	Description
full.h	42.26K	
Namespace “ns0” XML Schema Examples	302.17K	
enunciate-common.h	12.82K	Common header needed for all projects.
enunciate-common.m	42.34K	Common implementation code needed for all projects.

PHP Client Library

The PHP client-side library defines the PHP classes that can be (de)serialized to/from JSON. This is useful for accessing the REST endpoints that are published by this application, but only those that produce a JSON representation of their resources (content type "application/json").

This library requires the json_encode function which was included in PHP versions 5.2.0+.

Name	Source
json_encode	http://php.net/manual/en/function.json-encode.php

Files

The following files are applicable to this library:

Name	Size	Description
full.php	99.76K	

Ruby Client Library

The Ruby client-side library defines the Ruby classes that can be (de)serialized to/from JSON. This is useful for accessing the REST endpoints that are published by this application, but only those that produce a JSON representation of their resources (content type "application/json").

This library leverages the Ruby JSON Implementation, which is required in order to use this library.

Name	Source
Ruby JSON Implementation	http://json.rubyforge.org/

JSON REST Example

```
require 'net/https'
require 'uri'
//...

//read a resource from a REST url
url = URI.parse("...")
request = Net::HTTP::Post.new(url.request_uri)
input = Login.new
//set up the Login...
request.body = input.to_json
request['Content-Type'] = "application/json"

http = Net::HTTP.new(url.host, url.port)
//set up additional http stuff...
res = http.start do |ht|
  ht.request(request)
end

result = Login.from_json(JSON.parse(res.body))

//handle the result as needed...
```

Files

The following files are applicable to this library:

Name	Size	Description
full.rb	62.99	

Appendix

REST API Schemas

A

In this chapter

Files and Libraries Schemas

In this Topic

C Client Library

The following files are applicable to this library:

Name	Size	Description
full.c	387.11K	
enunciate-common.c	39.67K	Common code needed for all projects.

.NET Client Library

The following files are applicable to this library:

Name	Size	Description
full-dotnet.zip	6.77K	Contains full.cs file.
full.cs	122K	Contained in full-dotnet.zip file.

Objective C Client Library

The following files are applicable to this library:

Name	Size	Description
full.c	42.26K	
full.m	302.17K	

enunciate-common.h	12.82K	Common header needed for all projects.
enunciate-common.m	42.34K	Common implementation code needed for all projects.

Namespace “ns0” XML Schema Examples

The namespace “ns0” XML schema XSD examples are as follows:

WADL Example 1

```
Request URL:
https://localhost:8089/collector/events

Request Headers for XML output:
Authorization - Guest_1426270596791
Accept - Application/xml

Response body in XML:
&lt;?xml version="1.0" encoding="UTF-8" standalone="yes"?&gt;
&lt;events&gt;
&#x20;&#x20;&#x20;&lt;event&gt;
&#x20;&#x20;&#x20;&#x20;&#x20;&lt;description&gt;Profile Added/Updated
Successfully.&lt;/description&gt;
&#x20;&#x20;&#x20;&#x20;&#x20;&lt;category&gt;AUDIT&lt;/category&gt;
&#x20;&#x20;&#x20;&#x20;&#x20;&lt;messageID&gt;AUD-1005&lt;/messageID&g
t;
&#x20;&#x20;&#x20;&#x20;&#x20;&lt;timestamp&gt;1420842996921&lt;/timest
amp&gt;
&#x20;&#x20;&#x20;&#x20;&#x20;&lt;severity&gt;INFO&lt;/severity&gt;
&#x20;&#x20;&#x20;&#x20;&#x20;&lt;profileAction&gt;DROP&lt;/profileActi
on&gt;
&#x20;&#x20;&#x20;&#x20;&#x20;&lt;profileName&gt;NTP
Reflection&lt;/profileName&gt;
&#x20;&#x20;&#x20;&lt;/event&gt;
&lt;/events&gt;

Request Headers For JSON output:
Authorization - Administrator_1426270596791
Accept - Application/json

Response body in JSON:
{
"event":
[
{
"description": "Profile Added/Updated Successfully.",
"category": "Audit Event",
"messageID": "AUD-1005",
"timeStamp": 1420842996921,
"severity": "INFO",
"profileName": "NTP Reflection",
"profileAction": "NONE"
}
]
}
```

```
</pre>]]>
          </wadl:doc>
          <wadl:representation mediaType="application/xml" element="events"/>
          <wadl:representation mediaType="application/json"/>
        </wadl:response>
      </wadl:method>
      <wadl:method name="OPTIONS">
        <wadl:response>
          <wadl:representation mediaType="application/xml"/>
          <wadl:representation mediaType="application/json"/>
        </wadl:response>
      </wadl:method>
    </wadl:resource>
    <wadl:resource path="/collector/login">
      <wadl:method name="POST">
        <wadl:doc>
          <![CDATA[Allows user to login to the app using a valid username and
password.]]>
        </wadl:doc>
        <wadl:request>
          <wadl:doc>
            <![CDATA[- Login object containing valid username and password.]]>
          </wadl:doc>
          <wadl:representation mediaType="application/xml" element="login"/>
          <wadl:representation mediaType="application/json"/>
        </wadl:request>
        <wadl:response>
          <wadl:doc>
            <![CDATA[Login object - contains the token that can be used for
authorization of other REST requests.

<pre>
```

WADL Example 2

```
Request URL:
https://localhost:8089/collector/users

Request Headers for XML output:
Authorization - Guest_1426270596791
Accept - Application/xml

Response body in XML:
&lt;?xml version="1.0" encoding="UTF-8" standalone="yes"?&gt;
&lt;users&gt;
&#x20;&#x20;&#x20;&lt;user&gt;
&#x20;&#x20;&#x20;&#x20;&#x20;&#x20;&lt;name&gt;Administrator&lt;/name&gt;
&#x20;&#x20;&#x20;&#x20;&#x20;&#x20;&lt;access_privilege&gt;-1&lt;/access_pri
vilege&gt;
&#x20;&#x20;&#x20;&#x20;&#x20;&#x20;&lt;created_on&gt;Tue Dec 23 09:24:50 PST
2014&lt;/created_on&gt;
&#x20;&#x20;&#x20;&#x20;&#x20;&#x20;&lt;last_modified_on&gt;Tue Dec 23
09:24:50 PST 2014&lt;/last_modified_on&gt;
&#x20;&#x20;&#x20;&lt;/user&gt;
&#x20;&#x20;&#x20;&lt;user&gt;
&#x20;&#x20;&#x20;&#x20;&#x20;&#x20;&lt;name&gt;Guest&lt;/name&gt;
&#x20;&#x20;&#x20;&#x20;&#x20;&#x20;&lt;access_privilege&gt;0&lt;/access_priv
ilege&gt;
```

```
&#x20;&#x20;&#x20;&#x20;&#x20;&#x20;&lt;created_on&gt;Fri Mar 13 10:31:33 PDT
2015&lt;/created_on&gt;
&#x20;&#x20;&#x20;&#x20;&#x20;&#x20;&lt;last_modified_on&gt;Fri Mar 13
10:31:33 PDT 2015&lt;/last_modified_on&gt;
&#x20;&#x20;&#x20;&lt;/user&gt;
&lt;/users&gt;

Request Headers For JSON output:
Authorization - Administrator_1426270596791
Accept - Application/json

Response body in JSON:
{
"user": [
{
"name": "Administrator",
"access_privilege": -1,
"created_on": "Tue Dec 23 09:24:50 PST 2014",
"last_modified_on": "Tue Dec 23 09:24:50 PST 2014"
},
{
"name": "Guest",
"access_privilege": 0,
"created_on": "Fri Mar 13 10:31:33 PDT 2015",
"last_modified_on": "Fri Mar 13 10:31:33 PDT 2015"
}
]
}
</pre>]]>
          </wadl:doc>
          <wadl:representation mediaType="application/xml" element="users"/>
          <wadl:representation mediaType="application/json"/>
        </wadl:response>
      </wadl:method>
      <wadl:method name="POST">
        <wadl:doc>
          <![CDATA[Creates single new user. Only user with Admin privileges
(access_privilege=1) is allowed to create a new user. Valid name and
password are required as input. Access privilege is optional, if not
provided, created user will have operational privileges
(access_privilege=0). One cannot create user with same name.]]>
        </wadl:doc>
        <wadl:request>
          <wadl:doc>
            <![CDATA[- User object containing valid name and password and
optional
access_privilege.]]>
          </wadl:doc>
          <wadl:representation mediaType="application/xml"/>
          <wadl:representation mediaType="application/json"/>
        </wadl:request>
        <wadl:response>
          <wadl:doc>
            <![CDATA[<pre>
```

WADL Example 3

```
Request URL:
```

```
https://localhost:8089/collector/users

Request Headers For XML input:
Authorization - Administrator_1426270596791
Content-Type - Application/xml

Request payload in XML:
&lt;?xml version="1.0" encoding="UTF-8" standalone="yes"?&gt;
&lt;user&gt;
&#x20;&#x20;&#x20;&lt;name&gt;GuestOne&lt;/name&gt;
&#x20;&#x20;&#x20;&lt;password&gt;guest_one&lt;/password&gt;
&#x20;&#x20;&#x20;&lt;access_privilege&gt;0&lt;/access_privilege&gt;
&lt;/user&gt;

Request Headers For JSON input:
Authorization - Administrator_1426270596791
Content-Type - Application/json

Request payload in JSON:
{
"name": "GuestTwo",
"password": "guest_two",
"access_privilege": 1
}
</pre>]]>
          </wadl:doc>
          <wadl:representation mediaType="*/*"/>
          <wadl:representation mediaType="application/xml"/>
        </wadl:response>
      </wadl:method>
      <wadl:method name="OPTIONS">
        <wadl:response>
          <wadl:representation mediaType="application/xml"/>
          <wadl:representation mediaType="application/json"/>
        </wadl:response>
      </wadl:method>
    </wadl:resource>
    <wadl:resource path="/collector/users/{username}">
      <wadl:param name="username" style="template">
        <wadl:doc>
          <![CDATA[- the path parameter, which is the name of the user to be
updated.]]>
        </wadl:doc>
      </wadl:param>
      <wadl:method name="PUT">
        <wadl:doc>
          <![CDATA[Updates single user. Only user with Admin privileges
(access_privilege=1)
is allowed to update an existing user. Password and access_privilege can be
updated. User by name Administrator is root user and cannot be updated.]]>
        </wadl:doc>
        <wadl:request>
          <wadl:doc>
            <![CDATA[- the User object containing properties to be updated.]]>
          </wadl:doc>
          <wadl:representation mediaType="application/xml"/>
          <wadl:representation mediaType="application/json"/>
        </wadl:request>
        <wadl:response>
          <wadl:doc>
```

```
            <![CDATA[<pre>
```

WADL Example 4

```
Request URL:
https://localhost:8089/collector/users/GuestOne

Request Headers For XML input:
Authorization - Administrator_1426270596791
Content-Type - Application/xml

Request payload in XML:
&lt;?xml version="1.0" encoding="UTF-8" standalone="yes"?&gt;
&lt;user&gt;
&#x20;&#x20;&#x20;&lt;password&gt;guest_one_updated&lt;/password&gt;
&lt;/user&gt;

Request URL:
https://localhost:8089/collector/users/GuestTwo

Request Headers For JSON input:
Authorization - Administrator_1426270596791
Content-Type - Application/json

Request payload in JSON:
{
"password": "guest_two_updated",
"access_privilege": 0
}
</pre>]]>
          </wadl:doc>
          <wadl:representation mediaType="*/*"/>
          <wadl:representation mediaType="application/xml"/>
        </wadl:response>
      </wadl:method>
      <wadl:method name="DELETE">
        <wadl:doc>
          <![CDATA[Deletes single user. Only user with Admin privileges
(access_privilege=1)
is allowed to delete an existing user. User by name Administrator is root
user and cannot be deleted.]]>
        </wadl:doc>
        <wadl:request/>
        <wadl:response>
          <wadl:doc>
            <![CDATA[<pre>
```

WADL Example 5

```
Request URL:
https://localhost:8089/collector/users/GuestOne

Request Headers:
Authorization - Administrator_1426270596791

</pre>]]>
          </wadl:doc>
```

```
            <wadl:representation mediaType="*/*"/>
            <wadl:representation mediaType="application/xml"/>
          </wadl:response>
        </wadl:method>
      </wadl:resource>
    </wadl:resources>
  </wadl:application>
```

A Namespace “ns0” XML Schema Examples