PUBLIC

SAP BusinessObjects Business Intelligence platform
Document Version: 4.2 SP6 – 2018-07-18

Business Intelligence Platform RESTful Web Service Developer Guide

Content

1 Document History. 8

2 Getting Started. 11

3 Setting up the development environment. 12

3.1 Supported programming languages. 12

3.2 Using Ajax and JavaScript with RESTful web services across domains. 13

4 Using the SDK. 15

4.1 Retrieving the base URL for RESTful web service requests. 15

Retrieving the base URL through the CMC. 15

Retrieving the base URL programmatically. 15

4.2 Making RESTful web service requests. 16

Creating the request header. 21

Creating the request body. 22

Interpreting the response header. 22

Interpreting a response body in XML format. 25

Interpreting a response body in the JSON format. 29

Comparison of XML and JSON attributes. 35

Working with multilingual data. 38

4.3 Authentication. 38

To get a logon token from a user name and password. 39

To get a logon token from a serialized session or session token. 41

To get a logon token using an Active Directory Single Sign-On (AD SSO) account. 43

To get a logon token using trusted authentication. 44

Converting a logon token from XML-encoded text. 46

To add a logon token to a request header. 47

Using HTTP basic authentication. 48

To log off the BI platform. 49

Using authenticated sessions obtained from other SDKs. 50

SAML Configuration for REST end point. 52

4.4 Using logon tokens with OpenDocument URLs. 56

4.5 Navigating the BI platform repository. 58

To view the top level of the BI platform repository. 59

To retrieve an object by ID. 60

To retrieve an object by CUID. 61

To access child objects. 61

To use pagination with results. 63

To filter results by type. . . . 64
To access objects with relationships. . . . 65
4.6 Scheduling. . . . 68
To discover the scheduling URLs for an object. . . . 69
To schedule a resource. . . . 70
4.7 Configuring RESTful Web Services for Tomcat Server. . . . 72
4.8 Configuring RESTful Web Services for WebSphere Server. . . . 72

5 Administration and installation tasks. . . . 74
5.1 To install RESTful web services on Windows. . . . 74
5.2 To install RESTful web services on Unix. . . . 75
5.3 To configure web.xml to enable WinAD SSO. . . . 75
5.4 To configure Methods and Headers command line parameters. . . . 77
5.5 To configure the base URL for RESTful web services. . . . 78
5.6 To enable the error message stack. . . . 79
5.7 To set the default number of entries displayed on each page. . . . 79
5.8 To set the timeout value of a logon token. . . . 80
5.9 To configure session pool settings. . . . 80
5.10 To enable HTTP basic authentication. . . . 81
5.11 To configure cross-origin resource sharing (CORS) . . . 82
5.12 To enable and configure trusted authentication. . . . 82
5.13 Securing Microsoft Silverlight access to the WACS server. . . . 83

6 Configuring Recipient Delivery Rules. . . . 85

7 API Reference. . . . 86
7.1 About API Reference and Prerequisites. . . . 86
7.2 Fetching Information About the Build. . . . 88
7.3 Fetching the Timezone Information. . . . 89
7.4 Authentication. . . . 89
/v1/logon/long. . . . 89
/v1/logon/token. . . . 92
/v1/logon/adsso. . . . 95
/v1//logon/trusted. . . . 97
/v1//logon/trustedx509. . . . 100
/v1/logoff. . . . 101
7.5 Managing Scheduling. . . . 102
Getting Template for Scheduling. . . . 103
Creating Schedule now. . . . 105
Creating Schedule Once. . . . 107
Creating Hourly Schedule. . . . 110
Creating Daily Schedule. . . . 112

Creating Weekly Schedule. . . . 115
Creating Monthly Schedule. . . . 118
Creating Schedule for Nth Day of Month. . . . 120
Creating Schedule for First Monday of the Month. . . . 123
Creating Schedule for Calendar . . . 126
Creating Schedule for Xth Day of Nth Week of a Month . . . 128
Creating Schedule for Last Day of the Month. . . . 131
Getting Instances for a Report. . . . 134
Getting Instance Details of a Report. . . . 135
Getting Schedule List for a Report. . . . 138
Getting Instances Details of a Schedule. . . . 139
Sorting and Filtering. . . . 141
Getting Details of a Schedule. . . . 142
Finding Available Server Groups for Scheduling. . . . 144
Fetching Available Calendars and Calendar Dates. . . . 145
7.6 Managing Users. . . . 146
Creating New User. . . . 147
Getting User Details. . . . 150
Modifying User Details. . . . 151
Specifying Locale and Time-zone information for current user. . . . 153
Listing Available Locales and Timezones. . . . 155
Changing Password of Current User. . . . 157
Changing password of ABAP User. . . . 158
7.7 Deleting Users. . . . 160
7.8 Listing Users. . . . 161
7.9 Managing User Groups. . . . 165
Listing User Group. . . . 166
Creating New User Group. . . . 170
Getting User Group Details. . . . 172
Listing All Users in an User Group. . . . 173
Adding Users to a User Group. . . . 177
Removing Users from a User Group. . . . 178
Listing User Groups in a User Group. . . . 179
Adding User Groups to a User Group. . . . 183
Removing User Groups from a User Group. . . . 184
Modifying User Group Details. . . . 185
Deleting User Group. . . . 187
7.10 Managing Publications. . . . 188
Creating Publication. . . . 188
Listing Publications. . . . 190
Adding and Deleting Report Documents. . . . 194

Adding and Deleting Static Documents. . . . 196
Adding and Deleting Enterprise Users. . . . 198
Adding and Deleting Enterprise User Groups. . . . 201
Adding and Deleting Dynamic Recipients. . . . 203
Configuring Output Format for Reports. . . . 206
Configuring Destination Forms. . . . 208
Configuring Inbox Destination Plugin. . . . 209
Configuring SMTP Destination Plugin. . . . 211
Configuring FTP Destination Plugin. . . . 214
Configuring SFTP Destination Plugin. . . . 217
Configuring File System Destination Plugin. . . . 220
Configuring Target Object Name Per Document. . . . 222
Listing Schedule Instances of Publication. . . . 225
Getting Details of Publication and Modifying or Deleting Publication. . . . 226
Configuring Merged PDF Settings. . . . 229
Configuring the Scheduling Server Group. . . . 232
Configuring Notifications For Scheduled Publications. . . . 233
Configuring Advanced Settings. . . . 235
Configuring Events For Scheduled Publications. . . . 237
Configuring Delivery Rules and Conditions. . . . 239
Fetching Distribution Failure Information. . . . 246
Personalizing Publications. . . . 249
7.11 Uploading and Downloading File. . . . 259
/infostore/folder/<folder_id>/file. . . . 259
/infostore/folder/<SI_ID>/file. . . . 260
7.12 Managing Categories. . . . 262
Listing categories. . . . 262
Creating Categories. . . . 265
Getting Category Details. . . . 267
Modifying Category Details. . . . 268
Listing First Level Childern Under a Category. . . . 269
Listing the First Level Documents in a category. . . . 271
Deleting Categories. . . . 272
7.13 Managing Folders. . . . 273
Listing Folders. . . . 274
Creating a Folder. . . . 276
Getting Folder Details. . . . 278
Modifying folder details. . . . 279
Listing First Level Children Under a Folder. . . . 281
Uploading File to Folder. . . . 282
Deleting a Folder. . . . 283

7.14 Managing Documents. . . . 284
Listing Documents. . . . 284
Getting Document Details and the OpenDoc URL. . . . 287
Modifying Document Details. . . . 288
Getting Category Details of Document. . . . 290
Assigning a Document to Categories. . . . 291
Removing Categories from a Document. . . . 292
Sending Documents To Various Destinations. . . . 293
Deleting Documents. . . . 300
7.15 Managing Recycle Bin Items. . . . 300
Listing Recycle Bin Items. . . . 301
Retrieving Properties of Recycle Bin Items. . . . 302
Restoring Items from the Recycle Bin. . . . 303
Deleting Items from the Recycle Bin. . . . 304
7.16 Managing Alerts and Notifications. . . . 304
Listing Alerts for the Current User. . . . 305
Getting Details of an Alert. . . . 306
Editing Alerts. . . . 307
Deleting Alerts. . . . 308
Getting Notification Banner Details. . . . 309
Getting Count of Notification Banners. . . . 309
7.17 Infostore. . . . 310
Listing Objects In the Infostore. . . . 310
Getting Object Details. . . . 313
Listing Children of Objects. . . . 314
Lisitng Relationships of Objects. . . . 316
Getting Relationship Details Between Objects. . . . 318

8 CMS Query. . . . 320

9 BI Administrators' Cockpit. . . . 323
9.1 Server. . . . 323
Server Count. . . . 323
Server List. . . . 324
Server Details. . . . 329
Server Start . . . 330
9.2 Scheduled Jobs. . . . 331
Job Count. . . . 331
Job List. . . . 333
Job Details. . . . 336
Job Re-Run. . . . 337
Statistics Highest Instances. . . . 338

Statistics Longest Instances. 340
9.3 Content Usage. 342
Content Count. 343
Content List. 344
Statistics Folders by Reports. 348
Statistics Inbox by Unread. 350
Statistics Universe by Reports. 351
9.4 Applications. 353
Application Count. 353
Application List. 356

10 Appendix. 360
10.1 Appendix A - RWS error messages summary, categorized. 360
10.2 RESTful Web Services (RWS) Error Messages 364
RWS 00002 - RWS 00010. 365
RWS 00011 - RWS 00026. 368
RWS 00031 - RWS 00051. 373
RWS 00052 - RWS 00075. 380
RWS 00076 - RWS 00080. 389

1 Document History

The following table provides an overview of the most important document changes.

Version	Date	Description
SAP BusinessObjects Business Intelligence platform 4.1	May, 2013	First release of this document.
SAP BusinessObjects Business Intelligence platform 4.2	November, 2015	Added the following new sections to this guide: • Scheduling • Publication • User Management • User Group Management • Listing Schedule Instances of Publication • Uploading and Downloading Files • Generic APIs • cmsQuery
SAP BusinessObjects Business Intelligence platform 4.2 SP3	August, 2016	Added the following new sections with versioning vx=v1: • Infostore • Authentication • Scheduling • User Management • User Group Management • Publication • CMS Query Added the following new API sections: • About Information • Timezone Information • Personalizing Publication • Category Management • Folder Management • Document Management
SAP BusinessObjects Business Intelligence platform 4.2 SP4	March, 2016	Added the topic "About API Reference and Pre-requisites" within API reference Added the following **new sections** within API Reference:

Version	Date	Description
		• Managing Alerts and Notifications [page 304] • Managing Recycle Bin Items [page 300] Added the following **new** topics within API reference → Managing Documents • Assigning a Document to Categories [page 291] • Removing Categories from a Document [page 292] • Sending Documents To Various Destinations [page 293] **Updated** the following topic within API reference → Managing Documents • Getting Document Details and the OpenDoc URL Added the following **new** topic within API reference → Authentication • /v1//logon/trustedx509 Added the following **new** topics within API reference → Managing Publications • Configuring Merged PDF Settings [page 229] • Configuring the Scheduling Server Group [page 232] • Configuring Advanced Settings [page 235] • Configuring Notifications For Scheduled Publications [page 233] • Configuring Events For Scheduled Publications [page 237] • Configuring Delivery Rules and Conditions [page 239] • Fetching Distribution Failure Information [page 246] **Updated the sub-sections** of this topic within API reference → Managing Publications -> Personalizing Publications →Local Profiles [page 253] **Updated** the following topics within API reference → Managing Scheduling

Version	Date	Description
		• Getting Details of a Schedule [page 142] • Finding Available Server Groups for Scheduling [page 144] • Fetching Available Calendars and Calendar Dates [page 145] Added the following **new** topics within API reference → Managing Users • Specifying Locale and Time-zone information for current user [page 153] • Listing Available Locales and Timezones [page 155] • Changing Password of Current User [page 157]
SAP BusinessObjects Business Intelligence platform 4.2 SP5	September, 2017	Updated the following topics: • To configure the base URL for RESTful web services [page 78] (Added a caution in the procedure.) • Getting Instance Details of a Report [page 135](Added a foot-note in the topic.) Added the following topic: • Using the SDK → Configuring RESTful Web Services for Tomcat Server [page 72] • Using the SDK →Authentication →SAML Configuration for REST end point [page 52]
SAP BusinessObjects Business intelligence Platform 4.2 SP6	July 2018	- Updated the topic Configuring RESTful Web Services for Tomcat Server

2 Getting Started

The Business Intelligence platform RESTful web service SDK lets you access the BI platform using the HTTP protocol. You can use this SDK to log on to the BI platform, navigate the BI platform repository, access resources, and perform basic resource scheduling. You can access this SDK by writing applications that use any programming language that supports the HTTP protocol, or by using any tool that supports making HTTP requests. Both XML and JSON (JavaScript Object Notation) request and response formats are supported. For more information on the JSON format, see `www.json.org` and `tools.ietf.org/html/rfc4627`. For more information on the XML format, see `www.w3.org/XML/`

When to use this SDK

Use the RESTful web services SDK under the following conditions:

- You want to access BI platform repository objects or perform basic scheduling.
- You want to use a programming language that is not supported by other BI platform SDKs.
- You do not want to download and install BI platform libraries as a part of your application.

If you want to programmatically access the advanced functionality of the BI platform, including server administration, security configuration, and modifying the repository, use one of the BI platform SDKs that support these features. For example, use the SAP BusinessObjects Business Intelligence platform Java SDK, the SAP BusinessObjects Business Intelligence platform .NET SDK, or the SAP BusinessObjects Business Intelligence platform Web Services SDK to access the advanced features of the BI platform.

Start using the SDK

This guide is divided into these sections:

- Setting up the development environment - The supported development environments for using the RESTful web services SDK.
- Using the SDK - How to use the RESTful web services SDK, including how to log on to the BI platform, navigate the BI platform repository, schedule and access objects, interpret error messages, and work with multilingual content.
- Administration and installation tasks - How to install and configure RESTful web services on your BI platform deployment. This section is for BI platform system administrators.
- API Reference - A reference for RESTful web service requests.

3 Setting up the development environment

To develop applications that use the Business Intelligence platform RESTful web service SDK, you must be able to log on to a BI platform deployment that has RESTful web services installed, and know how to make HTTP requests:

- The BI platform deployment must have an instance of the RESTful web service installed and configured. Contact your BI platform administrator if RESTful web services are not installed and configured on your deployment.
- You must look up the base URL and port number that is used to listen for RESTful web service requests. You can find the base URL and port number either programmatically or by viewing it in the Central Management Console (CMC).
- You must be able to log on to the BI platform. You can do this by using a valid user ID and password, or by using a serialized session or session token that you have obtained from another SDK.
- You must know how to make HTTP requests, either by using your preferred programming language or a tool that supports making HTTP requests.

Related Information

Administration and installation tasks [page 74]
Authentication [page 38]
Retrieving the base URL for RESTful web service requests [page 15]

3.1 Supported programming languages

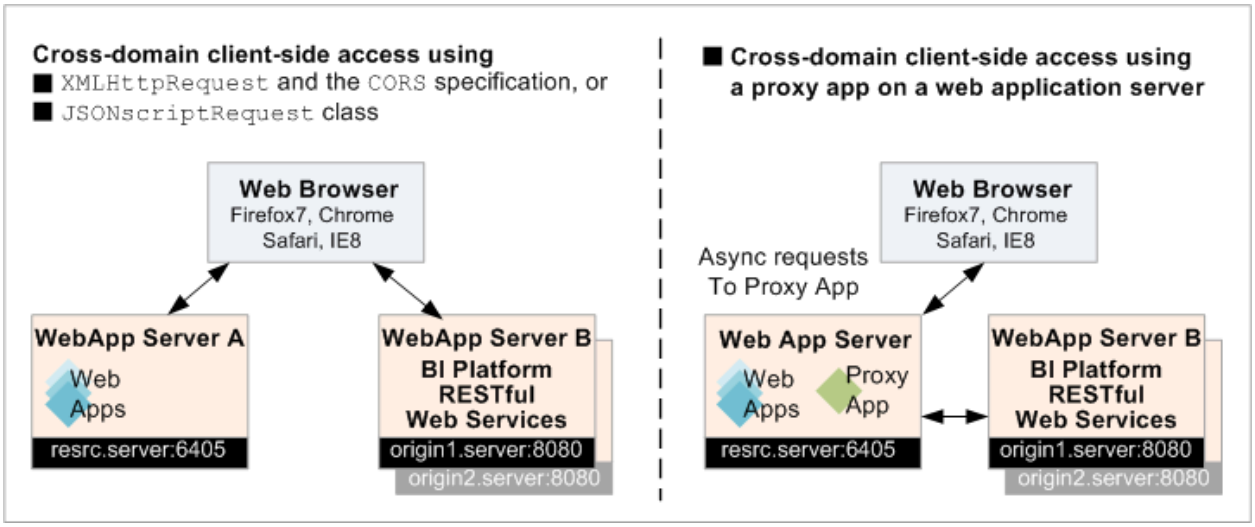
You can access the BI platform RESTful web services using any programming language that supports making HTTP requests. You are not required to include any libraries in your application.

Most advanced programming languages contain support for making HTTP requests. The cURL programming language has excellent support for HTTP requests, and provides cURL-based libraries for most major programming languages. For more information about cURL, see `http://curl.haxx.se/`.

You can also make HTTP requests without writing code by using tools that make HTTP requests. For example, you can obtain a REST Client plugin for Mozilla Firefox that allows you to make RESTful HTTP calls by specifying the URL, method, request header, and request body.

3.2 Using Ajax and JavaScript with RESTful web services across domains

Cross-domain HTTP requests to the BI platform RESTful web services are restricted by a security policy built into the JavaScript and Ajax languages. The intent is to prevent the operation of malicious scripts that may be unintentionally run from an untrusted server. This may include scripts hosted on different domains or from different ports on the same server or scripts hosted on the same server that uses a different protocol, for example `http` instead of `https`. Use one of the following workarounds to enable JavaScript or Ajax applications to make cross-domain requests to RESTful web services.



Using `XMLHttpRequest` and the `CORS` specification

When only client-side technologies are used such as HTML, CSS and JavaScript, cross-domain access is achieved by implementing `CORS` (Cross-Origin Resource Sharing) on the RESTful web server and the client-side web browser using the `XMLHttpRequest`. For more information on CORS, see `http://www.w3.org/TR/cors/`.

To restrict which domains may be accessed from the browser using CORS, the RESTful web server must be configured to include those domains.

Because various web browsers implement the CORS specification differently, use a library that allows the use of a single interface that works for all browsers and versions you intend to support.

> **Note**
>
> All of the Business Intelligence platform RESTful web service requests return results in XML or JSON format, so `XMLHttpRequest` may be used to process both response types. The JSONscriptRequest class is not restricted by the cross-origin requests.

Using a proxy app on a web application server

A proxy web application that runs on the same server as the JavaScript web page is used to forward HTTP requests for resources that exist on another server.

Proxies are used on websites that use server-side technologies such as JSP, Java Servlets, C# and ASP.NET. Web pages that use JavaScript and Ajax programming can make calls to other servers using a proxy application written in a programming language that does not have a same-origin security policy.

For security purposes, you can set up the proxy with suitable access restrictions to avoid unauthorized access to any internal or external networks. For example, if the required resources exist on domains `http://origin1.server:8080` and `http://origin2.server:8080`, the pass-through on the proxy server must only forward requests to only those addresses.

Related Information

To configure cross-origin resource sharing (CORS) [page 82]

4 Using the SDK

This section describes how to use the Business Intelligence platform RESTful web service SDK, including how to find the RESTful web services base URL, how to log on to the BI platform, and how to navigate the BI platform repository.

4.1 Retrieving the base URL for RESTful web service requests

To use the Business Intelligence platform RESTful web service SDK, you must know the protocol, server name, port number, and path of the service that listens to RESTful web service requests. Collectively, these form the base URL. Whenever you make a request to RESTful web services, the beginning of the request starts with the base URL and is followed by the specific details of the request.

Basic installations of the BI platform that are installed on a single server use the default base URL, `http://<servername>:6405/biprws/`.

In complex deployment scenarios, there can be multiple instances of the Web Application Container Server (WACS), which hosts the RESTful web service. In this case, RESTful web services may be hosted at a different location. The BI platform administrator defines the location base URL that is used to access RESTful web services, and you can discover the base URL programmatically or through the Central Management Console (CMC).

4.1.1 Retrieving the base URL through the CMC

You can find the base URL for RESTful web service requests by logging on to the Central Management Console (CMC) user interface and navigating to the RESTful web services setting.

1. Log on to the CMC.
2. Click *Applications*.
3. Right-click *RESTful Web Service* and click *Properties*.
   The *RESTful Web Service* properties window appears.
4. Retrieve the base URL from the *Access URL* text box.

4.1.2 Retrieving the base URL programmatically

You can programmatically discover the base URL for RESTful web services by using one of the other BI platform SDKs, for example the BI Platform Java SDK. To programmatically find the base URL for RESTful web services, you must first query the BI platform to retrieve the `SI_ACCESS_URL` property of the RESTful web service object. You can query for the RESTful web service object by its CUID, or by its kind. You can find the CUID and kind by accessing the Java constants,

`com.businessobjects.sdk.plugin.desktop.restwebservice.IRestWebService.CUID` and `com.businessobjects.sdk.plugin.desktop.restwebservice.IRestWebService.KIND`.

> **Note**
>
> The CUID value for RESTful web services is `AZpJlb9HDtxPjLHwEmF8xD8` and the kind value is `RestWebService`.

```
"SELECT SI_ACCESS_URL FROM CI_APPOBJECTS WHERE SI_CUID='" + IRestWebService.CUID +
"'"
```

```
"SELECT SI_ACCESS_URL FROM CI_APPOBJECTS WHERE SI_KIND=’" + IRestWebService.KIND +
"’"
```

Finding the base URL by using the BI platform Java SDK version 4.1

You can use the `getURL` method of the `IRestWebService` interface to retrieve the RESTful web services base URL.

```
IInfoObjects objects= infostore.query("SELECT SI_ACCESS_URL FROM CI_APPOBJECTS
WHERE SI_CUID='" + IRestWebService.CUID + "'");
IInfoObject object = (IInfoObject)objects.get(0);
IRestWebService restAppObject = (IRestWebService) object;
String baseUrl = restAppObject.getURL();
```

For more information on the BI platform Java SDK, see the *SAP BusinessObjects Business Intelligence Platform Java SDK Developer Guide*.

4.2 Making RESTful web service requests

To access the Business Intelligence platform RESTful web service SDK, you send HTTP requests to the URL that hosts the RESTful web services. The RESTful web service processes the request and returns a response that contains the requested information. You can access RESTful web services with any programming language or tool that supports HTTP requests. RESTful web services follow HTTP standards and the AtomPub specification, but also include custom attributes.

Requests consist of two main components, the request header and the request body. The request header defines the format of the request body, the accepted response format, and other custom settings such as the preferred language and the logon token. The request body may be left blank, or it may contain additional information needed to complete the request. For example, an authentication request passes the user name and password as formatted XML in the request body.

To make a RESTful web service request, you need the following:

- URL - The URL that hosts the RESTful web service.
- Method - The type of HTTP method to use for sending the request, for example `GET`, `PUT`, `POST`, or `DELETE`.

- Request header - The attributes that describe the request.
- Request body - Additional information that is used to process the request.

Once the request has been processed, you will receive a response. Responses contain the requested information, and include supporting information that you need to complete your next step. For example, responses may contain XML templates that can be used to populate the request body of subsequent requests, or they may contain links to related RESTful URLs, including parent folders, child folders, pages of additional information, and related links. By following the information provided by a RESTful response, you can navigate the requested data and obtain the templates you need in order to complete subsequent requests.

The Business Intelligence platform RESTful web service responses may be formatted as XML or JSON depending on the capabilities of the BI platform client application.

RESTful web service responses contain two main components:

- Response header - A list of attributes that describes the response format, and includes an HTTP response code.
- Response body - The requested information, and additional information that enables you to complete subsequent requests.

The examples in this document define the URL, method, request header attributes, and request body content that is required for each RESTful request. You can access the RESTful web services using any programming language or tool that supports HTTP requests.

Example

A RESTful POST request using the /logon/long API and response using the XML format

This example shows a RESTful request that logs on to the BI platform repository.

Request

URL: `http://localhost:6405/biprws/logon/long`

Method: `POST`

Request header attributes:

Attribute	Value
Content-Type	application/xml
Accept	application/xml

Request body:

```
<attrs xmlns="http://www.sap.com/rws/bip">
  <attr name="userName" type="string">username</attr>
  <attr name="password" type="string">password</attr>
  <attr name="auth" type="string"
possibilities="secEnterprise,secLDAP,secWinAD">secEnterprise</attr>
</attrs>
```

Response

Response header:

Attribute	Value
Status code	200 OK
Server	Apache-Coyote/1.1
X-SAP-LogonToken	"COMMANDCOM-LCM: 6400@{3&2=5542,U3&p=40680.8979564815,Y7&4F=12,U3&63=secEnterprise, 0P&66=60,03&68=secEnterprise:Administrator, 0P&qe=100,U3&vz=TZnoIE2yQyeLCkAlnHtaaYUHon5.p0yTkSaUiLC8SSM,UP}"
Date	Tue, 17 May 2011 21:33:03 GMT
Content-Type	application/xml
Content-Length	586

Response body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
  <author><name>@COMMANDCOM-LCM:6400</name></author>
  <id>tag:sap.com,2010:bip-rs/logon/long</id>
  <title type="text">Logon Result</title>
  <updated>2011-05-17T21:33:03.471Z</updated>
  <content type="application/xml">
    <attrs xmlns="http://www.sap.com/rws/bip">
      <attr name="logonToken" type="string">COMMANDCOM-LCM:6400@{3&amp;
2=5542,U3&am....YUHon5.p0yTkSaUiLC8SSM,UP}</attr>
    </attrs>
  </content>
</entry>
```

Example

A RESTful GET and POST request using the /logon/long API and response using the JSON format

This example shows a RESTful request that uses a `GET` request to retrieves a `JSON` formatted request body to use to enter the name and password and authentication type, then using a `POST` request to retrieve a logon token from the BI platform repository.

Request

URL: `http://localhost:6405/biprws/logon/long`

Method: `GET`

Request header attributes:

Attribute	Value
Accept	application/json

The request body that is returned in `JSON` format after a `GET` request appears as follows:

```
{"userName":"","password":"","auth":"secEnterprise"}
```

Request body that has an name label, for example `BOEuser` and password, for example `BOEPassword999` included before sending it as a `POST` request as showed in the following code snippet:

```
{"userName":"BOEuser","password":"BOEPassword999","auth":"secEnterprise"}
```

Note

The `auth` default value is `secEnterprise`. The authentication types that may be used include are as follows:

- `secEnterprise` - Enterprise authentication
- `secLDAP` - Lightweight Directory Access Protocol authentication
- `secWinAD` - Windows Active Directory authentication
- `secSAPR3` - SAP authentication

Response header after a `POST` request:

Attribute	Value
Status code	200 OK
Server	Apache-Coyote/1.1
X-SAP-LogonToken	:"COMMANDCOM-LCM: 6400@{3&2=5571,U3&p=40897.0049317824,Y7&4F=12,U3&63=secEnterprise, 0P&66=60,03&68=secEnterprise:BOEuser, 0P&qe=100,U3&vz=odiw9uLc1kVlJf9lggLFEWPAX3qsFWBT1LkdE2DTGhY,UP}"
Date	Tue, 17 December 2011 21:33:03 GMT
Content-Type	application/json
Content-Length	204

Response body in JSON format:

```
{"logonToken":"COMMANDCOM-LCM:
6400@{3&2=5571,U3&p=40897.0049317824,Y7&4F=12,U3&63=secEnterprise,
0P&66=60,03&68=secEnterprise:Administrator,0P&qe=100,U3&vz=odiw9uLc1kVlJf
9lggLFEWPAX3qsFWBT1LkdE2DTGhY,UP}"}
```

Example

A RESTful infostore JSON-formatted request

This example shows a RESTful request that uses a `GET` request and the `/infostore` API with a logon token to request information from BI platform repository that is returned in `JSON` format.

Request

URL: `http://commandcom-lcm:6405/biprws/infostore`

Method: `GET`

Request header attributes:

Attribute	Value
Accept	application/json
X-SAP-LogonToken	COMMANDCOM-LCM: 6400@{3&2=5542,U3&p=40680.897...UiLC8SS

Request body: (blank)

Response

Response header:

Attribute	Value
Status code	200 OK
Server	Apache-Coyote/1.1
X-SAP-LogonToken	COMMANDCOM-LCM: 6400@{3&2=5542,U3&p=40680.897...UiLC8SSM,UP}
Date	Tue, 17 December 2011 21:33:03 GMT
Content-Type	application/json
Content-Length	6919

Response body formatted as JSON. For clarity in the following code snippet, the back slash for escaped characters such as (/) and (") have been removed.


```
{
   "__metadata":
     {"uri":"http://localhost:9998/biprws/infostore/Root%20Folder/children?
page=1&amp;pageSize=3"},
   "first":
     {"__deferred":
     {"uri":"http://localhost:9998/biprws/infostore/Root%20Folder/children?
page=1&amp;pageSize=3"}
     },
   "next":
     {"__deferred":
     {"uri":"http://localhost:9998/biprws/infostore/Root%20Folder/children?
page=2&amp;pageSize=3"}
     },
   "last":
     {"__deferred":
     {"uri":"http://localhost:9998/biprws/infostore/Root%20Folder/children?
page=3&amp;pageSize=3"}
     },
   "entries":
   [
     {"__metadata":
       {"uri":"http://localhost:9998/biprws/infostore/4005"},
       "id":4005,
       "cuid":"FnKsrkkctAcA8BAAALB7kkQAADAFzVMX",
       "name":"Data Federation",
       "type":"Folder"
     },
     {"__metadata":
       {"uri":"http://localhost:9998/biprws/infostore/3931"},
       "id":3931,
       "cuid":"Ac1aKZlzj5VJmMQi5LDa53s",
       "name":"LCM",
       "type":"Folder"
     },
```

```
      {"__metadata":
        {"uri":"http://localhost:9998/biprws/infostore/5056"},
        "id":5056,
        "cuid":"Acu9FvxWBZ9Htt0_08a25b4",
        "description":"",
        "name":"Monitoring Report Sample",
        "type":"Folder"
       }
    ]
}
```

Related Information

Retrieving the base URL for RESTful web service requests [page 15]

4.2.1 Creating the request header

The request header of an HTTP request contains a set of attributes that describe the request. The BI platform RESTful web service SDK recognizes a set of standard HTTP attributes, as well as custom attributes defined specifically for the BI platform.

> **Note**
>
> The BI platform passes requests to other layers of the system, including client applications. You can include request header attributes that are not recognized by the BI platform but are recognized by client applications.

The following table describes request headers that are recognized by the BI platform:

Attribute	Description	Sample Value
`Content-Type`	The format of the request body. The BI platform accepts content of type `application/xml` or `application/json`. Client applications may accept other formats.	application/xml
`Accept`	The expected format of the response body. The BI platform provides content in the `application/xml` or in `application/json` format. Client applications may provide content in other formats.	application/json
`Accept-Language`	The preferred language used to retrieve system and error messages. This corresponds to the Product Locale (PL) of the BI platform.	en-US
`X-SAP-PVL`	The preferred language used to retrieve BI platform content. This corresponds	ja-JP

Attribute	Description	Sample Value
	to the Preferred Viewing Language (PVL).	
X-SAP-LogonToken	A logon token received from the authentication process. Enclose the logon token in quotation marks.	"COMMANDCOM-LCM: 6400@{3&2=55,3&p=403.0083,Y7&4F=12,U3&63=secEnterprise, 0P&56=60,03&68=secEnterprise:administrator, 0P&qe=100,U3&vz=y3EqvsvoVMU8raN2YjqDe4,UP}"
Authorization	The authorization type to use, for example HTTP basic authentication.	Basic <authtype> \<username>:<password> Replace <authtype> with the authentication type, <username> with your user name and <password> with the password.
X-SAP-TRUSTED-USER	The account name of a trusted user. The label X-SAP-TRUSTED-USER may be changed in CMC, *Servers List > WACS, Trusted Authentication Configuration* to another label such as MyUser.	trustedUser

Related Information

Authentication [page 38]

4.2.2 Creating the request body

The request body contains the information that RESTful web services needs to complete the request. For example, the request body of an authentication request contains the logon information, including user name and password. This provides the authentication URL with the information it needs to accept or reject the logon request.

You set an attribute in the request header to define the format of the request body. Set the Content-Type attribute in the message header to specify the format.

4.2.3 Interpreting the response header

The response header contains attributes that describe whether the request was successful, and describe the contents of the response body. Most of the response header attributes belong to the HTTP standard. However, the X-SAP-LogonToken header attribute is a custom attribute used only by the BI platform.

Status code

The status code contains a standard HTTP status code that describes whether the request was successful.

HTTP Response Code	Error	Description
400	Bad request	The requested resource exists, but the request contains errors.
401	Failed to logon or invalid session	Logon failed. Check that the username, password, and servername are correct.
403	Access denied	You do not have permission to operate on the requested resource. The current session may have expired. Log on to obtain a new session.
404	Service is not available	The requested service is not provided by the RESTful web services SDK.
405	Invalid request method	A request was made using a method that was not supported by the resource. For example, using a PUT request on a read-only resource.
406	Not acceptable	The requested resource cannot generate the content type specified by the `Accept` attribute of the request header.
408	BI platform server timeout	The server timed out waiting for the request.
415	Unsupported media type	The request contains a media type that the server or resource does not support.
500	RESTful web service internal error	An unclassified error occurred. See the response body for more information.
503	RESTful web service plugin not found	RESTful web services are not available. Verify that RESTful web services are configured correctly.

Server

The server that was used to process the request.

Date

The date and time of the response.

Content-Type

The format of the response body. For example, most web service responses use the value `application/xml` to show that the response body is formatted as XML.

Content-Length

The length of the response body.

Transfer-Encoding

The type of encoding that has been used to transport the message.

Content-Location

An alternative link that can be used to find the resource.

X-SAP-LogonToken

A token that can be used with subsequent requests to prove that you have been authenticated to access the BI platform. Authentication requests return the `X-SAP-LogonToken` custom attribute in the response header. Include the logon token in the request header of subsequent requests, and enclose it in quotation marks.

> **Note**
>
> A copy of the `X-SAP-LogonToken` value is returned in the response body of authentication responses. However, the response body is formatted as XML and converts the logon token to an XML-encoded version. This copy of the logon token must be converted back to its original format before it can be used.

Related Information

Converting a logon token from XML-encoded text [page 46]

4.2.4 Interpreting a response body in XML format

The Business Intelligence platform RESTful web service SDK provides responses in XML format, according to the Atom specification, available at *http://www.w3.org*. This section describes how XML tags apply to RESTful web services. The following screen illustrates how the BI launchpad returns XML data in response to a typical /infostore request.

```
<?xml version="1.0" ?>
<feed xmlns="http://www.w3.org/2005/Atom">
  <id>tag:sap.com,2010:bip-rs/infostore</id>
  <title type="text">InfoStore (@COMMANDCOM-LCM:6400)</title>
  <updated>2012-02-02T22:35:11.975Z</updated>
  <link href="http://localhost:6405/biprws/infostore/4/children?page=1&pageSize=5" rel="self"/>
  <link href="http://localhost:6405/biprws/infostore/4/children?page=1&pageSize=5" rel="first"/>
  <link href="http://localhost:6405/biprws/infostore/4/children?page=2&pageSize=5" rel="next"/>
  <link href="http://localhost:6405/biprws/infostore/4/children?page=7&pageSize=5" rel="last"/>
  <entry>
    <title type="text">Alert Notifications</title>
    <id>tag:sap.com,2010:bip-rs/ARZB.BFCQk9PqaqDpcFwo1w</id>
    <author>
      <name>System Account</name>
    </author>
    <link href="http://localhost:6405/biprws/infostore/Alert%20Notifications" rel="alternate"/>
    <content type="application/xml">
      <attrs xmlns="http://www.sap.com/rws/bip">
        <attr name="id"          type="int32">64</attr>
        <attr name="cuid"        type="string">ARZB.BFCQk9PqaqDpcFwo1w</attr>
        <attr name="description" type="string">Description here</attr>
        <attr name="name"        type="string">Alert Notifications</attr>
        <attr name="type"        type="string">Folder</attr>
      </attrs>
    </content>
  </entry>
  <entry>
    <title type="text">Application Folder</title>
      :
      :
    </content>
  </entry>
</feed>
```

<feed>

The `<feed>` element defines a list of `<entry>` elements. JSON uses curly brackets { and } to enclose a response.

```
<feed xmlns="http://www.w3.org/2005/Atom">
  <entry> ... </entry>
  <entry> ... </entry>
  ...
</feed>
```

<entry>

A single item. The `<entry>` tag may include the `xmlns` attribute.

```
<entry xmlns="http://www.w3.org/2005/Atom">
  ...
</entry>
```

<author>

The owner of the resource that was accessed. The `<author>` element includes a `<name>` element that defines the name of the owner of the resource. The following element shows that the owner of the resource is `System Account`.

```
<author><name>System Account</name></author>
```

<id>

A unique identifier of the resource.

```
<id>tag:sap.com,2010:bip-rs/AdoctK9h1sBHp3I6uG0Sh7M</id>
```

<title>

The name of the resource. This example shows that the name of the resource is `Application Folder`.

```
<title type="text">Application Folder</title>
```

<updated>

The date and time the resource was last updated.

```
<updated>2011-04-14T10:27:50.672Z</updated>
```

<link>

The `link` element defines links to URLs that can be used with other RESTful web service requests. These may include parent or child folders, or other information that is relevant to the request. By following these links, you can navigate through the BI platform repository.

The `href` attribute of the link tag defines the hyperlink, and the `rel` attribute describes the type of link. The following list describes possible values of the `rel` attribute:

<link> Related Attribute Name	Description
`self`	A link back to this URL.
`first`	A link to the first page of results.
`next`	A link to the next page of results.
`previous`	A link to the previous page of results.
`last`	A link to the last page of results.
`alternate`	Another link to the same resource.
`up`	A link to the parent of the current resource.
`related`	A link to a related resource.
`http://www.sap.com/rws/bip#children`	A link to the children of the current resource.
`http://www.sap.com/rws/bip#opendocument`	A link that can be used to view the resource with OpenDocument.
`http://www.sap.com/rws/bip#schedule`	A link that can be used to schedule a resource.

For example, the following link element describes a link to the next page of results:

```
<link href="http://localhost:6405/biprws/infostore/Root%20Folder/children?
page=3&pageSize=3" rel="next"></link>
```

Responses that provide links to documents also provide an OpenDocument URL that can be used to view documents using OpenDocument.

```
<link href="http://localhost:8080/BOE/OpenDocument/opendoc/openDocument.jsp?
sIDType=CUID&iDocID=Aa0U0jQbtKxCn.D3JDLOaHs" rel="http://www.sap.com/rws/
bip#opendocument" title="OpenDocument">
```

For more information about OpenDocument, see *Viewing Documents Using OpenDocument*.

> **Note**
>
> You can use logon tokens obtained from this SDK to authenticate with OpenDocument.

<content>

The payload of the RESTful response. The `<content>` element contains an `<attrs>` element, which itself contains a set of `<attr>` elements.

```
<content>
  <attrs>
    <attr>...</attr>
    <attr>...</attr>
  </attrs>
</content>
```

<attrs>

A list of properties of the content. The `<attrs>` element contains a set of `<attr>` elements.

```
<attrs>
  <attr>...</attr>
  <attr>...</attr>
</attrs>
```

<attr>

A property of the content.

Each `<attr>` element defines a property of the content. The `<attr>` tag uses two attributes, `name`, which describes the name of the property, and `type`, which describes the type of the property. The following example shows that the `id` property of the content is the value `43` (an integer), and the `name` property of the content is `Application Folder` (a string).

```
<attr name="id" type="int32">43</attr>
<attr name="name" type="string">Application Folder</attr>
```

This table describes the possible values for the `name` and `type` attributes of the `<attr>` tag.

Name	Type	Description
name	string	The name of the resource.
id	int32	The ID of the resource.
cuid	string	A unique identifier of the resource.
type	string	The type of resource, for example Folder or InfoView.
description	string	A description of the resource.
logonToken	string	A logon token.

<error>

Error codes.

Each <error_code> and <message> element refers to a RESTful Web Services error code reference in teh format RWS 000xx and includes a brief description. For more details, see the BusinessObjects XI *Error Messages Explained* guide.

4.2.5 Interpreting a response body in the JSON format

The Business Intelligence platform RESTful web service SDK provides responses in JSON format with the request header `accept : application/json`. This section describes how JSON tags apply to RESTful web services.

{ ... }

A JSON object is enclosed by curly brackets { and }, which is similar to the XML `<feed>` element.

```
{
    "__metadata": {
        "uri": "http://commandcom-lcm:6405/biprws/infostore/4/children?
page=1&pageSize=50"
    },
    "first": {
        "__deferred": {
            "uri": "http://commandcom-lcm:6405/biprws/infostore/4/children?
page=1&pageSize=50"
        }
    },
    "last": {
        "__deferred": {
            "uri": "http://commandcom-lcm:6405/biprws/infostore/4/children?
page=1&pageSize=50"
        }
    },
    "entries":
    [
        {
            "__metadata": {
                "uri": "commandcom-lcm:6405/biprws/infostore/Alert%20Notifications"
            },
            "id": 64,
            "cuid": "ARZB.BFCQk9PqaqDpcFwo1w",
            "name": "Alert Notifications",
            "type": "Folder",
            "uri":"alslsls"
        }
     .
     .
     .
    ]
}
```

"entries":

Entries are JSON objects within an array. The format is `"entries" : [{contentsOfEntryItem#1}, {contentsOfEntryItem#2}]`. The following example is a result of an `../infostore` RESTful Web Service API request. The `"entries":` part of the response shows two children named `"Alert Notifications"` and `"Users"`.

```
"entries":
    [
        {
            "__metadata": {
                "uri": "commandcom-lcm:6405/biprws/infostore/Alert%20Notifications"
            },
            "id": 64,
            "cuid": "ARZB.BFCQk9PqaqDpcFwo1w",
            "name": "Alert Notifications",
            "type": "Folder",
            "uri":"alslsls"
        },
     .
     .
     .
        {
            "__metadata": {
                "uri": "http://commandcom-lcm:6405/biprws/infostore/Users"
            },
            "id": 19,
            "cuid": "AXhmigik4CBKra9ZYzR2ezE",
            "description": "",
            "name": "Users",
            "type": "Folder"
        }
    ]
```

__metadata: { uri:

The `__metadata: { uri:` element equates to the XML <link> element. This defines links to URLs that can be used with other RESTful web service requests. These may include parent or child folders, or other information that is relevant to the request. By following these links, you can navigate through the BI platform repository.

The `href` attribute of the link tag defines the hyperlink, and the `rel` attribute describes the type of link. The following list describes possible values of the `rel` attribute. Note that the XML tags `alternate` and `related` have no JSON equivalent.

Attribute	Format	Example	Description
self	__metadata: { uri:	"__metadata": {"uri":"http://localhost:6405/biprws/infostore/4/children?page=1&pageSize=5"}	A link back to this URL.

Attribute	Format	Example	Description
first	first: { __deferred: { uri:	"first": {"__deferred": {"uri":"http://localhost:6405/biprws/infostore/4/children?page=1&pageSize=5"}	A link to the first page of results.
next	next: { __deferred: { uri:	"next": {"__deferred": {"uri":"http://localhost:6405/biprws/infostore/4/children?page=2&pageSize=5"}	A link to the next page of results.
previous	previous: { __deferred: { uri:	"previous": {"__deferred": {"uri":"http://localhost:6405/biprws/infostore/4/children?page=6&pageSize=5"}	A link to the previous page of results.
last	last: { __deferred: { uri:	"last": {"__deferred": {"uri":"http://localhost:6405/biprws/infostore/4/children?page=7&pageSize=5"}	A link to the last page of results.
up	up: { __deferred: { uri:	"up":{"__deferred": {"uri":"http://localhost:6405/biprws/infostore"}	A link to the parent of the current resource.
children	children: { __deferred: { uri:	"Children": {"__deferred": {"uri":"http://localhost:6405/biprws/infostore/User%20Folders/children"}	A link to the children of the current resource.
openDocument	opendocument { __deferred: { uri:	"openDocument": {"__deferred": {"uri":"http://commandcom-lcm:8080/BOE/OpenDocument/opendoc/openDocument.jsp?	A link that can be used to view the resource with OpenDocument.

Attribute	Format	Example	Description
		sIDType=CUID&iDocID=AQtkbbSqN4NOj3ydf.Sw1lY"}	
schedule	schedule { __deferred: { uri:	"Scheduling forms": {"__deferred": {"uri":"http://localhost:6405/biprws/infostore/4930/scheduleForms"} Use Post and include the schedule: "__metadata": {"uri":"http://localhost:6405/biprws/infostore/4930/scheduleForms/hourly"}	A link that can be used to schedule a resource. Use Get to retrieve the template, use Post to send the request.

For example, the following link element describes a link to the last page of results:

```
"last": {
        "__deferred": {
            "uri": "http://commandcom-lcm:6405/biprws/infostore/4/children?
page=1&pageSize=50"
        }
```

Responses that include document types, such as Web Intelligence and Crystal Reports, also provide an openDocument URL that can then be emailed or attached to a button control on a report.

In the following example, the `../infostore` API is used to retrieve the listing of a Web Intelligence openDocument-formatted links.

```
http://commandcom-lcm:6405/biprws/infostore/4930
```

```
{
   "up":{
      "__deferred":{
         "uri":"http://10.162.204.68:6405/biprws/infostore/4904"
      }
   },
   "Scheduling forms":{
      "__deferred":{
         "uri":"http://10.162.204.68:6405/biprws/infostore/4907/scheduleForms"
      }
   },
   "id":4907,
   "cuid":"AQtkbbSqN4NOj3ydf.Sw1lY",
   "openDocument":{
      "__deferred":{
         "uri":"http://commandcom-lcm:8080/BOE/OpenDocument/opendoc/
openDocument.jsp?
                sIDType=CUID&iDocID=AQtkbbSqN4NOj3ydf.Sw1lY"
      }
   },
   "description":"",
```

```
    "name":"Formatting Sample",
    "type":"Webi"
}
```

For more information about OpenDocument, see *Viewing Documents Using OpenDocument*.

> **Note**
>
> You can use the `../logon/long` API to obtain a logon token string that can be added to an openDocument URL so recipients do not have to provide their logon credentials.

Entry properties

Several properties make up the content of each entry item. The following example shows that the `id` property of the content is the value `64` (an integer), and the `name` property of the content is `Alert Notifications` (a string).

```
{
    "__metadata": {"uri": "commandcom-lcm:6405/biprws/infostore/Alert
%20Notifications"},
    "id": 64,
    "cuid": "ARZB.BFCQk9PqaqDpcFwo1w",
    "name": "Alert Notifications",
    "type": "Folder",
    "uri":"alslsls"
},
```

This table describes the available `name` and `type` properties for a JSON entry.

Name	Type	Example	Description
`name`	`string`	`"name": "Alert Notifications"`	The name of the resource.
`id`	`int32`	`"id": 64`	The ID number of the resource.
`cuid`	`string`	`"cuid": "ARZB.BFCQk9PqaqDpcFwo1w"`	A unique identifier of the resource.
`type`	`string`	`"type": "Folder"`	The type of resource, for example Folder or InfoView.
`description`	`string`	`"description": "Contains the ..."`	A description of the resource.
`logonToken`	`string`	`"type": "COMMANDCOM-LCM: 6400@{3&...Sv3b6vUJZe9...}"`	A logon token.
`uri`	`string`	`"uri": "http://localhost:6405/`	URI value.

Name	Type	Example	Description
		`biprws/infostore/Custom%20Roles"`	
`openDocument`	`string`	`"openDocument": {"__deferred": {"uri":"http://commandcom-lcm:8080/BOE/OpenDocument/opendoc/openDocument.jsp?sIDType=CUID&iDocID=AQtkbbSqN4NOj3ydf.Sw1lY"}`	An openDocument formatted URI value.

error_code

Each `error_code` and `message` element refers to a BI platform error or a RESTful Web Services error (RWS prefix) and includes a brief description. For more information, see the SAP BusinessObjects XI *Error Messages Explained* guide.

```
{
   "error_code":"FWM 01003",
   "message":"Server COMMANDC-OM-LCM:6400 not found or server may be down (FWM
01003)"
}
```

JSON escape characters

RESTful Web Services returns ASCII characters that are considered special by JSON by prefacing them with a back slash (`\`). The JSON specification for which characters must be escaped can be found at `http://www.ietf.org/rfc/rfc4627.txt` The following table lists several common ASCII++ characters that RESTful Web Service JSON requests will return prefaced with backslashes:

RWS - JSON	Unicode UTF-8	Description
`\b`	U+0008	Backspace
`\f`	U+000C	Form feed
`\n`	U+000A	New line
`\r`	U+000D	Carriage return
`\t`	U+0009	Tab
`\v`	U+000B	Vertical tab

RWS - JSON	Unicode UTF-8	Description
\'	U+0027	Single quote
\"	U+0022	Double quote
\\	U+005C	Back slash or reverse solidus
\/	U+005D	Forward slash or solidus
\u	U+xxxx	four-hex-digits

4.2.6 Comparison of XML and JSON attributes

RESTful Web Services requests that use XML always return some data to comply with the Atom specification. The following XML tags that do not have equivalents in the JSON data format, and it helps to be aware of them:

- <author>
- <id>
- <title>
- <updated>
- <link rel=alternate>
- <link rel=related>
- <content>
- <attrs>

Supported XML tags and JSON objects

The following table lists the XML tags and their equivalent JSON objects and entries supported by the BI platform RESTful Web Services implementation.

Supported XML tags and JSON objects

XML			JSON		Description
XML Tag	Sample	Type	Value	Type	
<feed>			{	JSON object	In a JSON result, the response is represented as a JSON Object. The XML <feed> tag equates to JSON's outermost curly brackets { }.
<entry>			entries : [{contentsOfEntryItem#1}, {contentsOfEntryItem#2}]		A request for a list of children, a collection of entries is returned, each one a JSON object. The collection of JSON objects is represented as an array in the "entries" name and value pair.

XML			JSON		Description
XML Tag	Sample	Type	Value	Type	
<author>			No JSON equivalent		These elements are not exposed in JSON.
<id>					
<title>					
<updated>					
<link>	rel=self		__metadata: { uri:		A link to your current location.
	rel=first		first: { __deferred: { uri::		A link to the first page of results.
	rel=next		next: { __deferred: { uri::		A link to the next page of results.
	rel=previous		previous: { __deferred: { uri::		A link to the previous page of results .
	rel=last		last: { __deferred: { uri::		A link to the last page of results .
	rel=alternate		No JSON equivalent.		An alternate link to your current location.
	rel=up		up: { __deferred: { uri::		A link to the parent of the current resource.
	rel=related		No JSON equivalent.		A link to a related resource.
	rel=http://www.sap.com/ rws/bip#children		children: { __deferred: { uri::		A link to the children of the current resource.
	rel=http://www.sap.com/ rws/bip#opendocument		opendocument: { __deferred: { uri::		A link that can be used to open a document such as a report or Adobe Acrobat PDF file.
	rel=http://www.sap.com/ rws/bip#schedule		schedule: { __deferred: { uri::		A link that can be used to schedule a resource.
<content>			No JSON equivalent.		For XML only, this is a container for the `<attrs>` element. `<content>` is required for the Atom feed specification, but not for JSON.
<attrs>					The XML element that contains one or more `<attr>` elements. In JSON, the attributes are presented as name and value pairs immediately within the JSON object representing the resource, rather than grouped as with the XML `<attrs>` tag.
<attr>	name=name	string	name:	JSON string	The name of the resource.
	name=id	int32	id:	JSON number	The numerical identification number of the resource.

XML			JSON		Description
XML Tag	Sample	Type	Value	Type	
	name=cuid	string	cuid:	JSON string	The 23 character alphanumeric cluster unique identifier.
	name=type	string	type:	JSON string	The type of resource, for example Folder or InfoView.
	name=description	string	description:	JSON string	The description of the resource.
	name=logonToken	string	logonToken:	JSON string	The logon token string.

Example

A comparison of XML and JSON format from an /infostore request

The following code snippet shows the hierarchy of RESTful Web Service elements with a typical /infostore `GET` request. On the left, is the XML listing. On the right, is the JSON listing of the same request. The corresponding lines of information are arranged for easier side-by-side comparison. To reduce the length of the code snippet, only the first object called "Alert Notifications" is shown. Note that this screenshot does not contain all available tags listed in the preceeding table.

XML

```
<?xml version="1.0" ?>
<feed xmlns="http://www.w3.org/2005/Atom">
  <id>tag:sap.com,2010:bip-rs/infostore</id>
  <title type="text">InfoStore (@COM...CM:6400)</title>
  <updated>2012-01-13T20:47:42.942Z</updated>
  <link href="http://...?page=1&pageSize=5" rel="self"/>
  <link href="http://...?page=1&pageSize=5" rel="first"/>
  <link href="http://...?page=6&pageSize=5" rel="previous"/>
  <link href="http://...?page=7&pageSize=5" rel="last"/>

  <entry>
    <title type="text">Alert Notifications</title>
    <id>tag:sap.com,2010:bip-rsARZB.BF...aqDpcFwo1w</id>
    <author><name>System Account</name></author>
    <link href="...infostore/Alert%20Notifications" rel="alternate"/>
    <content type="application/xml">
      <attrs xmlns="http://www.sap.com/rws/bip">
        <attr name="id" type="int32">64</attr>
        <attr name="cuid" type="string">ARZB.BF...pcFwo1w</attr>
        <attr name="description" null="true" type="string"/>
        <attr name="type" type="string">Folder</attr>
      </attrs>
    </content>
  </entry>
</feed>
```

JSON

```
{
  "__metadata":        {"uri": "http://...?page=1&pageSize=5"},
  "first":     {"__deferred": {"uri": "http://...?page=1&pageSize=5"}},
  "previous": {"__deferred": {"uri": "http://...?page=6&pageSize=5"}},
  "last":      {"__deferred": {"uri": "http://...?page=7&pageSize=5"}},
  "entries":
    [
      {
        "name": "Alert Notifications",
        {"__metadata": {"uri": "http://...infostore/Alert%20Notifications"},
        "id":  64,
        "cuid": "ARZB.BF...aqDpcFwo1w",
        "type": "Folder"
      }
    ]
}
```

4.2.7 Working with multilingual data

In multilingual environments, you can request the content and system messages to be returned in your preferred language. There are two request header attributes used to define the preferred language for content and system messages: `Accept-Language` and `X-SAP-PVL`.

When the BI platform software is installed, the user interface and system error messages are displayed in the Product Locale (PL). The available PL languages include the language packs that are installed with the BI platform software.

The system messages, including error messages, are returned in the language specified by the PL. You can request to use a specific language for system messages by setting the `Accept-Language` request header attribute. For example, to retrieve system messages in Japanese, set the `Accept-Language` request header attribute to `ja-JP`.

> **Note**
>
> If the requested PL is not available, the system messages are returned in the PL that was used when the BI platform software was installed.

The content in the BI platform may be stored in multiple languages. For example, the BI platform could store a report that has been translated into French, Japanese, and German. Use the `X-SAP-PVL` request header attribute to specify the preferred language of the content to be returned. If the content is not available in the requested language, it is returned in the closest available language. For example, to request content that is available in French, set the `X-SAP-PVL` request header attribute to `fr-FR`.

For more information about HTML language codes, see the HTML 4.01 specification at http://www.w3.org.

4.3 Authentication

To access the BI platform through the Business Intelligence platform RESTful web service SDK, you need a logon token. You get one by making a request to a logon URL. The token proves you have been authenticated as a valid user, and it can be included with subsequent RESTful web services requests without exposing sensitive information such as your password.

You can use any one of the following information types to obtain authentication and a resulting logon token:

- BI platform logon credentials. This method supports WinAD, LDAP, SAP and Enterprise authentication. For more information about authentication, see the *SAP BusinessObjects Business Intelligence Platform Administrator Guide*.
- A session token from another session. If you have access to a session that has already been authenticated, you can use the session token to obtain a logon token.

  > **Note**
  >
  > The session token obtained from another SDK is not the same as the logon token, and cannot be used directly with RESTful web service requests.

- A serialized session. If you have access to a session that has already been authenticated, you can use it to obtain a logon token.

If your authentication request was successful, the response header includes a logon token. This logon token is defined by `X-SAP-LogonToken`.

> **Note**
>
> The response body contains a copy of the logon token. However, this copy of the logon token is embedded in XML and has converted (encoded) illegal XML characters, such as `&`, < and > to an XML-friendly format. You must convert the XML encoded characters back to their original format before you can use this copy of the logon token. Alternatively, you can use the copy of the logon token that is provided in the response header, which has not been formatted for XML.

Each time you make a request to RESTful web services, you must add the `X-SAP-LogonToken` attribute to the request header, and set its value to be the logon token you received from being authenticated. Enclose the logon token in quotation marks, because it may contain characters that are not otherwise allowed in the request header.

The following table contains an example of a logon token:

Attribute	Sample Value
`X-SAP-LogonToken`	`"COMMANDCOM-LCM: 6400@{3&2=5604,U3&p=40623.9446463889,Y7&4F=12,U3&63=secEnterprise, 0P&68=secEnterprise:Administrator, 0P&qe=100,U3&vz=g5KUV8cAA.d_ARmSDnBy6T7jJVNyFCTso4s0q3dI.4k,UP}"`

Related Information

Converting a logon token from XML-encoded text [page 46]

4.3.1 To get a logon token from a user name and password

Before you can log on to the BI platform, you must have retrieved the base URL for RESTful web service requests.

To log on to the BI platform and obtain a logon token, make a request to `http://<baseURL>/logon/long` using the `POST` method, providing your user name, password, and type of authentication in the request body.

You can use the following types of authentication to log on to the BI platform:

- WinAD
- LDAP
- SAP
- Enterprise

To discover how to format the body of the logon request, make a request to the same URL, `http://<baseURL>/logon/long`, using the `GET` method. This response contains an XML template that can be used to format the request body of the logon request. The XML template includes a list of the supported authentication types.

1. Create a new HTTP request.
2. Add the `Accept` attribute to the request header, and set its value to `application/xml`.
3. Use the `GET` method to send the request to the `http://<baseURL>/logon/long` URL.

   Replace `<baseURL>` with the base URL for RESTful web services.

   ```
   GET http://localhost:6405/biprws/logon/long
   ```

   The response body contains a template.

   ```
   <attrs xmlns="http://www.sap.com/rws/bip">
     <attr name="userName" type="string"/></attr>
     <attr name="password" type="string"></attr>
     <attr name="auth" type="string"
   possibilities="secEnterprise,secLDAP,secWinAD,secSAPR3">secEnterprise</attr>
   </attrs>
   ```

4. Create a new HTTP request.
5. Add the `Accept` attribute to the request header, and set its value to `application/xml`.
6. Add the `Content-Type` attribute to the request header, and set its value to `application/xml`.
7. Fill out the XML template with the user name, password, and authentication type, and add it to the request body of the new request.

   ```
   <attrs xmlns="http://www.sap.com/rws/bip">
     <attr name="userName" type="string">myUserName</attr>
     <attr name="password" type="string">myPassword</attr>
     <attr name="auth" type="string"
   possibilities="secEnterprise,secLDAP,secWinAD,secSAPR3">secEnterprise</attr>
   </attrs>
   ```

8. Use the `POST` method to send the request to the same URL, `http://<baseURL>/logon/long`.

   Replace `<baseURL>` with the base URL for RESTful web services.

   ```
   POST http://localhost:6405/biprws/logon/long
   ```

   The response header returns the logon token as the `X-SAP-LogonToken` attribute.

   ```
   X-SAP-LogonToken:"COMMANDCOM-LCM:
   6400@{3&2=5595,U3&p=40674.9596541551,Y7&4F=12,U3&63=secEnterprise,
   0P&66=60,03&68=secEnterprise:Administrator,
   0P&qe=100,U3&vz=SFY6agrLPxpfQBK1ZKYCwoBZKCbfsQm7VgWZFiH.RhM,UP"
   ```

   The logon token is contained between the quotation marks. In the example above, the logon token is as follows:

   ```
   COMMANDCOM-LCM:6400@{3&2=5595,U3&p=40674.9596541551,Y7&4F=12,U3&63=secEnterprise,
   0P&66=60,03&68=secEnterprise:Administrator,
   0P&qe=100,U3&vz=SFY6agrLPxpfQBK1ZKYCwoBZKCbfsQm7VgWZFiH.RhM,UP
   ```

   The response body contains a copy of the logon token in the `<attr>` element. If the logon token contains characters that are illegal in XML, they are replaced with their XML-encoded value. For example the `&` character is replaced with `&`. To use a logon token taken from the response body, you must convert the XML-encoded logon token back to its original format.

   The following example shows how the XML-encoded logon token appears in the response body:

   ```
   <entry xmlns="http://www.w3.org/2005/Atom">
   ```

```
  <author><name>@COMMANDCOM-LCM:6400</name></author>
  <id>tag:sap.com,2010:bip-rs/logon/long</id>
  <title type="text">Logon Result</title>
  <updated>2011-03-07T20:48:56.015Z</updated>
  <content type="application/xml">
    <attrs xmlns="http://www.sap.com/rws/bip">
      <attr name="logonToken" type="string">COMMANDCOM-LCM:6400@{3&amp;
2=5595,U3&amp;p=40674.9596541551,Y7&amp;4F=12,U3&amp;63=secEnterprise,0P&amp;
66=60,03&amp;68=secEnterprise:Administrator,
0P&amp;qe=100,U3&amp;vz=SFY6agrLPxpfQBK1ZKYCwoBZKCbfsQm7VgWZFiH.RhM,UP}</attr>
    </attrs>
  </content>
</entry>
```

Related Information

Retrieving the base URL for RESTful web service requests [page 15]
Converting a logon token from XML-encoded text [page 46]

4.3.2 To get a logon token from a serialized session or session token

To log on with this method, you must be able to use another BI platform SDK to access an existing authenticated session, for example, use the BI platform Java SDK. You must also know the base URL for RESTful web service requests.

You can get a logon token for RESTful web services from a valid session token or a serialized session. Make a request to the `http://<baseURL>/logon/token` URL using the `POST` method, and provide an XML-encoded version of the serialized session or session token in the request body. Replace `<baseURL>` with the base URL for RESTful web services.

To discover how to format the request body, make a request to the same URL, `http://<baseURL>/logon/token` using the `GET` method. The response from this request contains an XML template that can be used with the request body of the logon request.

By using a serialized session to obtain a logon token, you do not increase the number of concurrent user licenses used by the BI platform. However, using a session token will increase the concurrent user license count by one.

1. Create a new HTTP request.
2. Use the `GET` method to send the request to the `http://<baseURL>/logon/token` URL.

   Replace `<baseURL>` with the base URL for RESTful web services.

   ```
   GET http://localhost:6405/biprws/logon/token
   ```

   The response contains an XML template.

   ```
   <attrs xmlns="http://www.sap.com/rws/bip">
     <attr name="tokenType" type="string" possibilities="token,
   serializedSession">token</attr>
     <attr name="logonToken" type="string" null="true"></attr>
   </attrs>
   ```

3. Create a new HTTP request.
4. Add the `Content-Type` attribute to the request header, and set its value to `application/xml`.
5. Fill out the XML template and add it to the request body.

   Set the value of the `<attr name ="tokenType" type="string">` element to be `token` if you are using a session token, and set it to `serializedSession` if you are using a serialized session. Set the value of the `<attr name="logonToken" type="string">` element to an XML-encoded version of the serialized session or session token value.

```
<attrs xmlns="http://www.sap.com/rws/bip">
  <attr name="tokenType" type="string" possibilities="token,
serializedSession">serializedSession</attr>
  <attr name="logonToken" type="string">3&amp;ua=AWmaEx4Z.NVPpAEthuTGAjc,
8P&amp;ub=AfRWaT5_131NlLLf5bRMLKY,8P&amp;S5,88&amp;
5U=5320JaqlNvF1mr4m8u5UQFadItj5319JWKkfBwlKLBfrgXC8Npg1jC,8P&amp;
63=secEnterprise,8P&amp;2r=COMMANDCOM-LCM:6400,8P&amp;3k=@COMMANDCOM-LCM:
6400,8P&amp;1=Administrator account,8P&amp;W={},?z&amp;
4E=5319JWKkfBwlKLBfrgXC8Npg1jC,8P&amp;Tn={3&amp;.1={3&amp;
2=726,03&amp;O=FavoritesFolder,0P},2z&amp;.2={3&amp;
2=727,03&amp;O=PersonalCategory,0P},2z&amp;.3={3&amp;2=728,03&amp;O=Inbox,0P},
2z&amp;U=3,03},?z&amp;4F=12,8P&amp;Tm=36500,83&amp;uy=-1043,8L&amp;
35=Administrator,8P&amp;ux=AeiCInd_R6lBrV98duvX1dc,8P&amp;pa,8P</attr>
</attrs>
```

> **Note**
>
> This example shows a serialized session. The serialized session or session token value must be XML-encoded to remove illegal XML characters. For example, replace the `&` character with `&`.

6. Use the `POST` method to send the request to the same URL, `http://<baseURL>/logon/token`. Replace `<baseURL>` with the base URL for RESTful web services.

```
POST http://localhost:6405/biprws/logon/token
```

   The response header returns the logon token as the `X-SAP-LogonToken` attribute.

```
X-SAP-LogonToken:"COMMANDCOM-LCM:
6400@{3&2=5595,U3&p=40674.9596541551,Y7&4F=12,U3&63=secEnterprise,
0P&66=60,03&68=secEnterprise:Administrator,
0P&qe=100,U3&vz=SFY6agrLPxpfQBK1ZKYCwoBZKCbfsQm7VgWZFiH.RhM,UP"
```

   The logon token is contained between the quotation marks.

> **Note**
>
> The response body contains a copy of the logon token in the `<attr>` element. If the logon token contains characters that are illegal in XML, they are replaced with their XML-encoded value. For example, the `&` character is replaced with `&`. To use a logon token taken from the response body, you must convert the XML-encoded logon token back to its original format.
>
> The following example shows how the XML-encoded logon token appears in the response body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
  <author><name>@COMMANDCOM-LCM:6400</name></author>
  <id>tag:sap.com,2010:bip-rs/logon/token</id>
  <title type="text">Logon Result</title>
  <updated>2011-06-28T17:54:31.994Z</updated>
  <content type="application/xml">
```

```
    <attrs xmlns="http://www.sap.com/rws/bip">
      <attr name="logonToken" type="string">COMMANDCOM-LCM:6400@{3&amp;
2=5319,U3&amp;p=40722.7462034491,Y7&amp;4F=12,U3&amp;63=secEnterprise,0P&amp;
66=60,03&amp;68=secEnterprise:Administrator,
0P&amp;qe=100,U3&amp;vz=KeDu7064jWSptBT_m5BkBJ5Q_NaxyvE_WStqXmigYrg,UP}</attr>
    </attrs>
  </content>
</entry>
```

Related Information

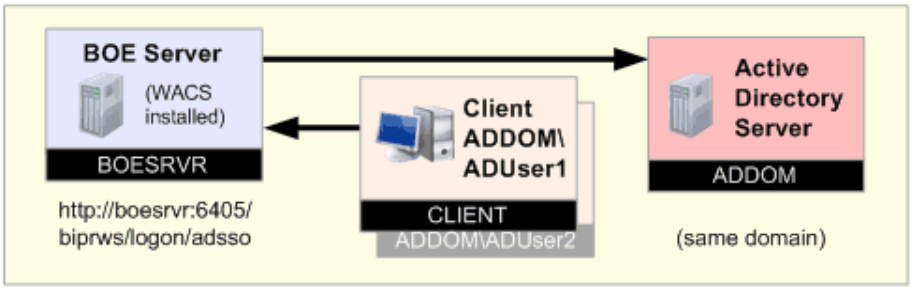
Retrieving the base URL for RESTful web service requests [page 15]

Using authenticated sessions obtained from other SDKs [page 50]

Converting a logon token from XML-encoded text [page 46]

4.3.3 To get a logon token using an Active Directory Single Sign-On (AD SSO) account

To use the Active Directory Single Sign-On feature of RESTful Web Services, clients must have a Windows Active Directory (WinAD) account and be logged into the computer that will be using the `/logon/adsso` API. Clients must also have logon accounts on the BI platform that match the WinAD accounts. The following diagram illustrates the configuration and authentication relationship between the BI platform server, the client computer, and the Windows Active Directory server.



Once the WinAD SSO feature is enabled as described in `Administration and installation tasks > To configure web.xml to enable WinAD SSO`, clients can use their WinAD credentials to log on to their computer. Those credentials will be used to authenticate them for access to the BI platform server automatically.

Use the following steps to obtain a logon token through AD SSO.

1. Create a new HTTP request.
2. Use the `GET` method to send the request to `http://<baseURL>/logon/adsso`.

   Replace `<baseURL>` with the base URL for RESTful web services.

For example:

```
GET http://localhost:6405/biprws/logon/adsso
```

The response header returns the logon token as the `X-SAP-LogonToken` attribute. An example XML response appears as follows:

```
<?xml version="1.0" ?>
<entry xmlns="http://www.w3.org/2005/Atom">
  <author>
    <name>
      @BOESRVR.ADDOM.COM
    </name>
  </author>
  <id>
    tag:sap.com,2010:bip-rs/logon/adsso
  </id>
  <title type="text">
    Logon Result
  </title>
  <updated>
    2011-11-11T11:11:11.340Z
  </updated>
  <content type="application/xml">
    <attrs xmlns="http://www.sap.com/rws/bip">
      <attr name="logonToken" type="string">
        BOESRVR.ADDOM.COM:6400@{3&2=4584,U3&p=40868.9276775116,Y7&4F=4331,U3
        &63=secWinAD,0P&66=60,03&68=secWinAD:CN%3DADUser1%2CCN%3DUsers%2CDC%3D
        ADDOM%2CDC%3DCOM,0P&qe=100,U3&vz=
        kOox8TDqAiFsfs8T3GefI3sWXIyKymc9qvytAjihC7w,UP}
      </attr>
    </attrs>
  </content>
</entry>
```

3. Use the resulting `X-SAP-LogonToken` within an HTTP request header to make further RESTful Web Service requests (for example `http://<baseURL>/infostore`.) You can also HTTP-encode the logon token and append it to an OpenDocument URL with the `&token=<logonToken>` parameter.

Related Information

Retrieving the base URL for RESTful web service requests [page 15]
Converting a logon token from XML-encoded text [page 46]

4.3.4 To get a logon token using trusted authentication

To use the trusted authentication feature of RESTful Web Services, the features must be activated as described in *Administration and Installation tasks > To enable and configure trusted authentication*.

Trusted authentication is used to speed up access to protected resources once users have already been authenticated elsewhere; for example, after users have logged in with a Windows account.

The methods of logon token retrieval, using trusted authentication, are as follows:

- HTTP header requests using a customizable header for the user name.
- URL queries.
- Cookie authentication.

To use one of the three trusted authentication logon retrieval methods, open CMC and go to *WACS* > *Trusted Authentication Configuration*, in the *Retrieving Method* menu, change the option to match the method you will be using. For all trusted authentication methods, there is an option to change the *Name Parameter*, which is found in *Servers* > *Core Services* > *WACS*. Note that all URLs and values supplied are case sensitive.

Retrieving Method	RESTful API used	Usage instructions
`HTTP_HEADER`	/logon/trusted	1. Create an HTTP request using the `GET` method. 2. Use the `/logon/trusted` API, for example, `http://localhost:6405/biprws/logon/trusted` 3. Create a request header with the default label `X-SAP-TRUSTED-USER`, and add a trusted user name, for example `bob`. The resulting logon token is displayed in the response header.
`QUERY_STRING`	/logon/trusted?<MyUser>=<Username>	1. In a web browser URL, use the `/logon/trusted` API, and add the user name parameter and the user name, for example, `http://localhost:6405/biprws/logon/trusted?MyUser=bob`. For example: ○ Replace `MyUser` with a customized user name parameter that is set in CMC under *Servers > Core Services > WACS > Trusted Authentication Configuration*. ○ Replace `bob` with a that of a trusted user that is set in CMC under *Users and Groups* > *User List*. The resulting logon token is displayed in the browser body window.
`COOKIE`	/logon/trusted	1. Create a cookie, and add the following information: ○ The domain. For example, `localhost`. ○ The name label, for example the default value of `X-SAP-TRUSTED-USER`, with the

Retrieving Method	RESTful API used	Usage instructions
		value for the logon name, for example, `bob`. ○ The path, for example `/` (forward slash). 2. Enter the URL, for example, `http://localhost:6405/biprws/logon/trusted` and press the *Enter* key to see the resulting logon token displayed in the browser window.

Related Information

Retrieving the base URL for RESTful web service requests [page 15]
Converting a logon token from XML-encoded text [page 46]

4.3.5 Converting a logon token from XML-encoded text

Logon tokens are returned in both the response header and the response body of authentication responses. The response body is formatted as XML, which reserves certain characters for its own use. If the logon token contains these characters, they are replaced with character sequences that are allowed to be embedded in XML but will not work in a logon token. Before you can use an XML-encoded logon token, it must be converted back to its original format.

> **Note**
>
> You only need to perform this step if you retrieve the logon token from the response body. The logon token that is contained in the response header is not XML-encoded.

To convert an XML-encoded logon token to its original format, replace each XML-encoded character sequence with the character it represents. For example, replace the `&` character encoding with the `&` character.

The following table shows the examples of the most common XML encoding of illegal XML characters.

XML encoding	Character
`'`	'
`"`	"
`&`	&
`<`	<
`>`	>

For more information about representing characters in XML, refer to the specification for extensible markup language at *http://www.w3.org*.

Example

This example shows a XML-encoded logon token.

```
COMMANDCOM-LCM:6400@{3&amp;2=5675,U3&amp;p=40653.0083064583,Y7&amp;4F=12,U3&amp;
63=secEnterprise,0P&amp;66=60,03&amp;68=secEnterprise:Administrator,
0P&amp;qe=100,U3&amp;vz=y3EqvsvoehahHhbmPrpaPjKVMU8raN3zEpnt2YjqDe4,UP}
```

The example shows the logon token after it has been converted to its original format.

```
COMMANDCOM-LCM:6400@{3&2=5675,U3&p=40653.0083064583,Y7&4F=12,U3&63=secEnterprise,
0P&66=60,03&68=secEnterprise:Administrator,
0P&qe=100,U3&vz=y3EqvsvoehahHhbmPrpaPjKVMU8raN3zEpnt2YjqDe4,UP}
```

4.3.6 To add a logon token to a request header

Once you have obtained a logon token, you can use it to authenticate RESTful requests that access the BI platform.

Note

If you obtained the logon token from the request body, you must convert it from its XML-encoded format back to its original format. Alternatively, you can obtain the original logon token directly from the response header.

For example, this text represents a logon token that is embedded in the XML of a response body.

```
COMMANDCOM-LCM:6400@{3&amp;2=5675,U3&amp;p=40653.0083064583,Y7&amp;4F=12,U3&amp;
63=secEnterprise,0P&amp;66=60,03&amp;68=secEnterprise:Administrator,
0P&amp;qe=100,U3&amp;vz=y3EqvsvoehahHhbmPrpaPjKVMU8raN3zEpnt2YjqDe4,UP}
```

This text represents a logon token obtained for a response header, or a token obtained from a response body that has been converted back to its original format.

```
COMMANDCOM-LCM:6400@{3&2=5675,U3&p=40653.0083064583,Y7&4F=12,U3&63=secEnterprise,
0P&66=60,03&68=secEnterprise:Administrator,
0P&qe=100,U3&vz=y3EqvsvoehahHhbmPrpaPjKVMU8raN3zEpnt2YjqDe4,UP}
```

1. Create a new RESTful web service request or modify an existing request.
2. Add an attribute to the request header.
3. Set the name of the attribute to `X-SAP-LogonToken`.
4. Set the value of the attribute to the logon token value, and enclose the value in quotation marks.

Name	Value
X-SAP-LogonToken	"COMMANDCOM-LCM:6400@{3&2=5604,U3&p=40623.9456463889,Y7&4F=12,U3&63=secEnterprise,0P&68=secEnterprise:Administrator,0P&qe=100,U3&vz=g5KUU8cAA.d_ARmSDnBy6T7jJVNyFCTso4s0q3dI.4k,UP}"

Related Information

Converting a logon token from XML-encoded text [page 46]

4.3.7 Using HTTP basic authentication

Use HTTP basic authentication to log on to the BI platform without including a logon token in the HTTP header of the RESTful web service request. Instead, you provide your user name, password, and an authentication type.

> **Note**
>
> User names and passwords are not transmitted securely using HTTP basic authentication, unless they are used in conjunction with HTTPS.

HTTP basic authentication must be enabled by an administrator. The administrator may also define a default authentication type that is used if you do not specify an authentication type.

Authentication types

You can use the following authentication types with HTTP basic authentication:

- `secEnterprise` - Enterprise authentication
- `secLDAP` - LDAP authentication
- `secWinAD` - Windows AD authentication
- `secSAPR3` - SAP authentication

Making requests using HTTP authentication consumes a license. If session caching is not used, a license is consumed for the duration of the request and is released once the request is completed. If session caching is used, the license associated with the cached session is used.

> **Note**
>
> The user name, password, and authentication type must be base64-encoded as defined by RFC 2716. User names that contain the `:` character cannot be used with HTTP basic authentication.

Using HTTP basic authentication in a web browser

To log on with a web browser using the default authentication type, provide your user name and password at the prompt.

To log on using a particular authentication type, use `<authenticationType>\<username>` in the user name field, and provide your password in the password prompt. Replace `<authenticationType>` with the type of authentication, and `<username>` with your user name. For example, to log on using SAP authentication with the

user name `myUserName`, enter `secSAPR3\myUserName` in the user name field, and enter your password in the password field.

> **Note**
>
> Sessions do not persist across multiple browser requests. Each service request you make through the basic authentication needs a username and password encoded in base64 format in the request header.

Using HTTP basic authentication programmatically

To use HTTP basic authentication programmatically, add the `Authorization` attribute to the request header, and set its value to be the base64-encoded value of the authorization string.

Use the following authorization string to use the default authentication type:

```
Basic <username>:<password>
```

Use the following authorization string to use a specific authentication type:

```
Basic <authtype>\<username>:<password>
```

4.3.8 To log off the BI platform

Before you can log off the BI platform, you must know the base URL for RESTful web service requests. You also must have the logon token for the session that you want to invalidate.

Logon tokens expire automatically if they are not used for a set time. By default, logon tokens expire after one hour of inactivity, but this value can be configured by an administrator. To log off of your session before it expires automatically, make a `POST` request to the `http://<baseURL>/logoff` URL. Replace `<baseURL>` with the base URL for RESTful web services.

By logging off the BI platform, you invalidate the logon token and release any license that is associated with the session.

1. Create a new HTTP request.
2. Add the `Accept` attribute to the request header, and set its value to `application/xml`.
3. Add the `X-SAP-LogonToken` attribute to the request header, and set its value to the logon token value, enclosed in quotation marks.

Name	Value
X-SAP-LogonToken	"COMMANDCOM-LCM:6400@{3&2=5604,U3&p=40623.9456463889,Y7&4F=12,U3&63=secEnterprise,0P&68=secEnterprise:Administrator,0P&qe=100,U3&vz=g5KUU8cAA.d_ARmSDnBy6T7jJVNyFCTso4s0q3dI.4k,UP}"

4. Use the `POST` method to send the request to the `http://<baseURL>/logoff` URL.

   Replace `<baseURL>` with the base URL for RESTful web services.

   ```
   POST http://<baseURL>/logoff
   ```

   If the logoff attempt was successful, the response header contains the HTTP status code 200.

Related Information

Retrieving the base URL for RESTful web service requests [page 15]

4.3.9 Using authenticated sessions obtained from other SDKs

You can use another BI platform SDK to obtain a serialized session or session token from an existing authenticated session. You can then obtain a logon token for the Business Intelligence platform RESTful web service SDK by providing the serialized session or session token in a request to the `/logon/token` URL.

You can use serialized sessions or session tokens obtained from the following SDKs, version XI 3.0 and later:

- SAP BusinessObjects Business Intelligence platform Java SDK
- SAP BusinessObjects Business Intelligence platform .NET SDK
- SAP BusinessObjects Business Intelligence platform Web Services SDK

4.3.9.1 Getting session information with the BI platform Java SDK

You can use the BI platform Java SDK to obtain a serialized session or session token from an existing session that has already been authenticated. Provide the serialized session or session token in the body of a request to the `/logon/token` URL to obtain a logon token for the Business Intelligence platform RESTful web service SDK.

To get a serialized session, use the `getSerializedSession` method of the `IEnterpriseSession` class.

```
ISessionMgr sessionMgr = CrystalEnterprise.getSessionMgr();
IEnterpriseSession enterpriseSession = sessionMgr.logon("username", "password",
"cmsname", "secEnterprise");
String serializedSession = enterpriseSession.getSerializedSession();
```

To get a session token, use the `getDefaultToken` or the `createLogonToken` method of the `ILogonTokenMgr` class.

```
ISessionMgr sessionMgr = CrystalEnterprise.getSessionMgr();
IEnterpriseSession enterpriseSession = sessionMgr.logon("username", "password",
"cmsname", "secEnterprise");
String sessionToken = enterpriseSession.getLogonTokenMgr().getDefaultToken();
```

For more information about using the BI platform Java SDK, see the *SAP BusinessObjects Business Intelligence Platform Java SDK Developer Guide*.

4.3.9.2 Getting session information with the BI platform .NET SDK

You can use the BI platform .NET SDK to obtain a serialized session or session token from an existing session that has already been authenticated. Provide the serialized session or session token in the body of a request to the `/logon/token` URL to obtain a logon token for the Business Intelligence platform RESTful web service SDK.

To get a serialized session, use the `SerializedSession` property of the `EnterpriseSession` class.

```
SessionMgr sessionMgr = new SessionMgr();
EnterpriseSession session = sessionMgr.Logon("username", "password", "cms",
"secEnterprise");
string serializedSession = session.SerializedSession;
```

To get a session token, use the `SerializedSesion` property or the `CreateLogonTokenEx` method of the `LogonTokenMgr` class.

```
SessionMgr sessionMgr = new SessionMgr();
EnterpriseSession session = sessionMgr.Logon("username", "password", "cms",
"secEnterprise");
string logonTokenMgr = session.LogonTokenMgr.DefaultToken;
```

4.3.9.3 Getting session information with the BI platform Web Services SDK

You can use the BI platform Web Services SDK to obtain a serialized session or session token from an existing session that has already been authenticated. Provide the serialized session or session token in the body of a request to the `/logon/token` URL to obtain a logon token for Business Intelligence platform RESTful web service SDK.

To get a serialized session, use the `getSerializedSession` method of the `SessionInfo` class.

```
URL boConURL = new URL("http://boserver:port/dswsbobje/services/Session");
Connection connection = new Connection(boConURL);
Session session = new Session(connection);
EnterpriseCredential credential = EnterpriseCredential.Factory.newInstance();
credential.setLogin("username");
credential.setPassword("password");
credential.setDomain("domain");
credential.setAuthType("secEnterprise");
SessionInfo sessionInfo = session.login(credential);
String serializedSession = sessionInfo.getSerializedSession();
```

To get a session token, use the `getDefaultToken` method of the `SessionInfo` class.

```
URL boConURL = new URL("http://boserver:port/dswsbobje/services/Session");
Connection connection = new Connection(boConURL);
Session session = new Session(connection);
EnterpriseCredential credential = EnterpriseCredential.Factory.newInstance();
credential.setLogin("username");
credential.setPassword("password");
credential.setDomain("domain");
credential.setAuthType("secEnterprise");
SessionInfo sessionInfo = session.login(credential);
```

```
String sessionToken = sessionInfo.getDefaultToken();
```

For more information about using the BI platform Web Services Consumer Java SDK, see the *SAP BusinessObjects Business Intelligence Platform Web Services Consumer Java SDK Developer Guide*.

4.3.10 SAML Configuration for REST end point

Any client that mimics the behavior of a web browser for SAML Authentication can invoke SAML REST API. In SAML API, SAML logon (http://host:<port>/biprws/v1/logon/saml), takes no query parameters. It reads the user principal from the HTTP request object and enables trusted authentication for that user principal, and returns the serialized token.

To configure SAML for REST end point, follow the procedure below:

1. To configure Trusted Authentication for Tomcat with *Websession* as the option in REST, modify the file in the location: `<INSTALLDIR>\SAP BusinessObjects\tomcat\webapps\biprws\WEB-INF\config\custom\biprws.properties` as shown in the screenshot below:

```
#-------------------------------------------------------Default CMS Configuration-------------------------------------------------------
CMS_Default=TARGETMACHINE:6400
#-------------------------------------------------------System Property Configuration-------------------------------------------------------
Default_Number_Of_Objects_On_One_Page=50
Enterprise_Session_Token_Timeout_In_Minutes=60
Session_Pool_Size=1000
Session_Pool_Timeout_In_Minutes=2
#-------------------------------------------------------Cross-Origin Resource Sharing Configuration-------------------------------------------------------
Allow_Origins=*
Max_Age_In_Minutes=1440
#-------------------------------------------------------Logger properties-------------------------------------------------------
LogLevel=High
#-------------------------------------------------------Trusted Authentication Configuration-------------------------------------------------------
Retrieving_Method=HTTP_HEADER
User_Name_Parameter=X-SAP-TRUSTED-USER
Trusted_Auth_Shared_Secret=
#-------------------------------------------------------Active Directory Properties-------------------------------------------------------
Krb5_Ini_File_Location=
bscLogin_Conf_File_Location=
```

2. Create the IDP user in the BI Platform or import through SDK script, or export using the CSV option in the CMC.If you are using SAP Cloud Platform Identity Provider, export all the users and then import them to the BI platform. Refer How to import users in bulk from Central Management Console.
3. Activate the XML tags in the *web.xml* file and delete the `work`folder.
   1. Go to `<INSTALLDIR>\tomcat\webapps\biprws\WEB-INF`.
   2. Open *web.xml*.
   3. Search for **filter** in the `web.xml` file.

```
        </servlet-mapping>
    <!--SAMl-->
<!--<filter>
        <filter-name>springSecurityFilterChain</filter-name>
        <filter-class>
org.springframework.web.filter.DelegatingFilterProxy</filter-
class>
    </filter>
```

4. Uncomment `filter` to enable SAML.

```
        </servlet-mapping>
    <!--SAMl-->
<filter>
        <filter-name>springSecurityFilterChain</filter-name>
        <filter-class>
org.springframework.web.filter.DelegatingFilterProxy</filter-
class>
    </filter>
```

5. Search for **filter-mapping** in the `web.xml` file.

```
<filter-mapping>
        <filter-name>springSecurityFilterChain</filter-name>
        <url-pattern>/*</url-pattern>
           <dispatcher>REQUEST</dispatcher>
        <dispatcher>FORWARD</dispatcher>
    </filter-mapping>-->
```

6. Uncomment **filter-mapping** to enable SAML.

```
<filter-mapping>
        <filter-name>springSecurityFilterChain</filter-name>
        <url-pattern>/*</url-pattern>
           <dispatcher>REQUEST</dispatcher>
        <dispatcher>FORWARD</dispatcher>
    </filter-mapping>
```

7. Similarly, search for **context-param** and **listener**.
8. Uncomment **context-param** and **listener**.

```
        <!--SAML-->
    <!--<context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>
            /WEB-INF/securityContext.xml
        </param-value>
    </context-param>-->
     <!--SAML-->
     <!--SAML-->
    <!--<listener>
    <listener-class>
org.springframework.web.context.ContextLoaderListener</listener-
class>
</listener>-->
```

9. Save the file.
10. Delete the `work` folder under biprws in `<BOE Install Dir>\tomcat\work`.

4. To update the IDP metadata in the Service Provider, download the IDP metadata from the respective IDP service providers. Copy the metadata file to `<BOE Install Dir>\tomcat\webapps\biprws\WEB-INF` and rename it to **idp-meta-downloaded.xml** . For more details on downloading the IDP metadata, refer Tenant SAML 2.0 Configuration
5. Restart the Tomcat application server.

If BOE is deployed on any non -Windows machine, the path seperators in filepath to the IDP metadata under **FilesystemMetadataProvider** should be changed in `securityContext.xml` under `<BOE Install Dir>\tomcat\webapps\biprws\WEB-INF`. That is,

Sample Code

```
<value type="java.io.File">/WEB-INF/idp-meta-downloaded.xml</value> has to be
changed to <value type="java.io.File">\WEB-INF\idp-meta-downloaded.xml</value>
for non-Windows
<bean
class="org.springframework.security.saml.metadata.ExtendedMetadataDelegate">
<constructor-arg>
<bean class="org.opensaml.saml2.metadata.provider.FilesystemMetadataProvider">
<!-- URL containing the metadata -->
<constructor-arg>
<!-- <value type="java.io.File">file:///C:/idp-meta-downloaded.xml</value> -->
<!-- <value type="java.io.File">/WEB-INF/idp-meta-downloaded.xml</value> --
><!-- For Windows -->
<value type="java.io.File">\WEB-INF\idp-meta-downloaded.xml</value> <!-- For
non-Windows -->
</constructor-arg>
<property name="parserPool" ref="parserPool"/>
</bean>
```

It appears as shown in the screenshot below:

Here, HCP is used as the IDP. The downloaded IDP metadata file is as shown in the screenshot.

Note

- This is a one-time activity.
- Only one IDP is supported at a time for authentication.

6. (Optional) Generate keystore for enabling SAML 2.0.

   This step is applicable only if you want to use your own keystore file.

   SAML exchanges involve usage of cryptography for signing and encryption of data. A sample self-signed keystore `sampletestKeystore.jks` is packaged with the product and is valid till October 18, 2019. `sampletestKeystore.jks` has an alias name `Testkey` and password `Password1`. You can now generate a self-signed keystore file using the JAVA utility keytool. Follow the steps below to generate a keystore file:

   1. Navigate to `<INSTALLDIR>\SAP BusinessObjects Enterprise XI 4.0\win64_x64\sapjvm\bin`.
   2. Run the command: `keytool -genkeypair -alias aliasname -keypass password -keystore samplekeystore.jks -validity numberofdays`

Command	Description
-alias	Enter the alias name of the certificate
-keypass	Enter the certificate's password
-keystore	Name of the keystore file
-validity	Validity of the certificate
numberofdays	Number of days for which the self-signed certificate is valid.

The following questions are prompted after executing the command:

- Enter keystore password: *****
- Re-enter new password: *****
- What is your first and last name? : `<MY_FIRST_AND_LAST_NAME>`
- What is the name of your organizational unit? : `<MY_ORGANIZATIONAL_UNIT>`
- What is the name of your organization? : `<MY_ORGANIZATION>`
- What is the name of your city and locality? : `<MY_CITY>`
- What is the name of your State and Province? : `<MY_STATE>`
- What is the two-letter country code for this unit? : `<COUNTRY_CODE>`

3. Stop the Tomcat application server.
   The keystore file is generated at `<INSTALLDIR>\SAP BusinessObjects Enterprise XI 4.0\win64_x64\sapjvm\bin`.
4. Move the keystore file to `<INSTALLDIR>\tomcat\webapps\biprws\WEB-INF`.
5. Edit the `securityContext.xml` file located at `<INSTALLDIR>\tomcat\webapps\biprws\WEB-INF` with the new alias name, password, and keystore file name. Refer the XML code below:

Sample Code

```
<bean id="keyManager"
class="org.springframework.security.saml.key.JKSKeyManager">
<constructor-arg value="/WEB-INF/sampleKeystore.jks"/>
<constructor-arg type="java.lang.String" value="Password1"/>
<constructor-arg>
<map>
<entry key="aliasname" value="password"/>
</map>
</constructor-arg>
<constructor-arg type="java.lang.String" value="Testkey"/>
</bean>
```

Refer the table below for understanding the arguments:

XML Tag	Description
`<constructor-arg value="/WEB-INF/sampleKeystore.jks"/>`	Locates the keystore file.

XML Tag	Description
`<constructor-arg type="java.lang.String" value="Password1"/>`	Password for the keystore file.
`<entry key="aliasname" value="password"/>`	Alias password.
`<constructor-arg type="java.lang.String" value="Testkey"/>`	Alias of the default certificate

6. Restart Tomcat.

> **Note**
>
> SP metadata must be generated every time the keystore file is changed.

7. Generate and upload the service provider metadata.
   a. Go to: http://host:tomcatport/biprws/v1/logon/saml/saml/metadata. This automatically downloads an XML file.
   b. Upload the XML to the identity provider metadata.

> **Note**
>
> - A pre-generated service Provider (SP) metadata file is shipped by default. You can edit and upload the same metadata file. In `<BOE Install Dir>\tomcat\webapps\biprws\WEB-INF spring_saml_metadata.xml`, replace the XML tags `<replace_withip>` with the IP address or hostname of the machine depending on your network, and `<replace_withport>` with the Tomcat port number.
> - As an example for HCP as IDP, refer the steps below:
> 1. After SAML application is created in HCP, upload the SP metadata.
> 2. Create a new application under *Applications*.
> 3. Upload the SP metadata.

4.4 Using logon tokens with OpenDocument URLs

OpenDocument syntax allows you to create hyperlinks that directly link to documents stored in the BI platform. The Business Intelligence platform RESTful web services SDK provides some support for working with OpenDocument. Logon tokens obtained from the Business Intelligence platform RESTful web services SDK can be used to authenticate with OpenDocument, and some RESTful responses return OpenDocument links.

For more information about using OpenDocument, see *Viewing Documents Using OpenDocument*.

Obtaining OpenDocument links for documents

When you request a document, for example a Crystal report or a Webl report, the response includes a OpenDocument link that can be used to view the resource with OpenDocument.

Links to OpenDocument URLs can be identified by the `rel` attribute, `"http://www.sap.com/rws/bip#opendocument"`, and the `title` attribute, `OpenDocument`.

```
<link href="http://localhost:8080/BOE/OpenDocument/opendoc/openDocument.jsp?
sIDType=CUID&iDocID=Aa0U0jQbtKxCn.D3JDLOaHs" rel="http://www.sap.com/rws/
bip#opendocument" title="OpenDocument”>
```

Appending the logon token to the OpenDocument URL

You can authenticate an OpenDocument URL by appending a logon token obtained using the Business Intelligence platform RESTful web services SDK to the end of the URL.

The syntax of the logon token parameter is shown below. Replace `<openDocumentURL>` with the OpenDocument URL and replace `<logonToken>` with the URL-encoded logon token value.

```
<openDocumentURL>&token=<logonToken>
```

> **Note**
>
> A URL-encoded logon token may contain a large number of characters. Some web browsers may limit the number of characters that are allowed in a URL.

The following example shows how to add a logon token to the end of the OpenDocument URL, `http://localhost:8080/BOE/OpenDocument/opendoc/openDocument.jsp?sIDType=CUID&iDocID=AYymBvuJZTRAlkojmuUj36w`.

1. Get a logon token by authenticating with the BI platform RESTful web services SDK.

   ```
   COMMANDCOM-LCM:6400@{3&2=5521,U3&p=40709.9614065046,Y7&4F=12,U3&63=secEnterprise,
   0P&66=60,03&68=secEnterprise:Administrator,
   0P&qe=100,U3&vz=sIQcJghbp2BvJrgPBNGJrRruBpfSShro9.ipdnKzqXM,UP}
   ```

2. URL-encode the logon token.

   ```
   COMMANDCOM-LCM%3A6400%40%7B3%262%3D5521%2CU3%26p%3D40709.9614065046%2CY7%264F
   %3D12%2CU3%2663%3DsecEnterprise%2C0P%2666%3D60%2C03%2668%3DsecEnterprise
   %3AAdministrator%2C0P%26qe%3D100%2CU3%26vz
   %3DsIQcJghbp2BvJrgPBNGJrRruBpfSShro9.ipdnKzqXM%2CUP%7D
   ```

   > **Note**
   >
   > There are many free tools available that can URL-encode strings.

3. Append `&token=<logonToken>` to the end of the OpenDocument URL. Replace `<logonToken>` with the URL-encoded logon token.

   ```
   http://localhost:8080/BOE/OpenDocument/opendoc/openDocument.jsp?
   sIDType=CUID&iDocID=AYymBvuJZTRAlkojmuUj36w&token=COMMANDCOM-LCM
   ```

```
%3A6400%40%7B3%262%3D5521%2CU3%26p%3D40709.9614065046%2CY7%264F
%3D12%2CU3%2663%3DsecEnterprise%2C0P%2666%3D60%2C03%2668%3DsecEnterprise
%3AAdministrator%2C0P%26qe%3D100%2CU3%26vz
%3DsIQcJghbp2BvJrgPBNGJrRruBpfSShro9.ipdnKzqXM%2CUP%7D
```

Adding the logon token to the OpenDocument request header

You can add the `X-SAP-LogonToken` attribute to the HTTP request header of an OpenDocument request, and set its value to be the value of the logon token. Enclose the logon token in quotation marks. Add the logon token to the request header when you want to avoid URL-encoding the logon token and appending a large number of characters to the end of the OpenDocument URL.

Name	Value
X-SAP-LogonToken	"COMMANDCOM-LCM: 6400@{3&2=5604,U3&p=40623.9456463889,Y7&4F=12,U3&63=secEnterprise, 0P&68=secEnterprise:Administrator, 0P&qe=100,U3&vz=g5KUU8cAA.d_ARmSDnBy6T7jJVNyFCTso4s0q3dI.4k,UP}"

Related Information

Authentication [page 38]

4.5 Navigating the BI platform repository

You can navigate through the BI platform repository, also known as the InfoStore, by requesting objects and following the links provided by the responses. Responses contain links to parent folders, child objects, and other related information. For example, when you request a folder, the response contains a link that returns the children of the folder. You can also retrieve objects directly by requesting them by their ID or CUID.

You can limit the number of entries returned by a response by requesting objects of a certain type, or by splitting a large number of entries across multiple pages.

Before you can view the contents of the BI platform repository, you must be authenticated and have obtained a logon token. Pass the logon token in the request header of each request by adding the `X-SAP-LogonToken` attribute to the request header and setting its value to be the logon token.

4.5.1 To view the top level of the BI platform repository

Before you can view the BI platform repository, you must have obtained a valid logon token and know the base URL for RESTful web service requests.

You can make a request to view the top level of the BI platform repository, also known as the InfoStore. The returned result contains links that you can follow to navigate child folders and explore the repository.

1. Create a new HTTP request.
2. Add the `X-SAP-LogonToken` attribute to the request header, and set its value to be a valid logon token.
3. Use the `GET` method to send the request to the `http://<baseURL>/infostore/` URL.

   Replace `<baseURL>` with the base URL for RESTful web service requests.

   ```
   GET http://localhost:6405/biprws/infostore
   ```

4. The response contains a feed that contains links children and entries that describe the top-level folders of the repository.

   Each `<link>` entry contains a hyperlink to a RESTful URL that can be used to access the resource directly. The list of attributes contains properties of the resource.

   ```
   <feed xmlns="http://www.w3.org/2005/Atom">
     <id>tag:sap.com,2010:bip-rs/infostore</id>
     <title type="text">InfoStore (@COMMANDCOM-LCM:6400)</title>
     <updated>2011-03-31T23:55:10.852Z</updated>
     <link href="http://localhost:6405/biprws/infostore/4/children?
   page=1&amp;pageSize=50" rel="self"></link>
     <link href="http://localhost:6405/biprws/infostore/4/children?
   page=1&amp;pageSize=50" rel="first"></link>
     <link href="http://localhost:6405/biprws/infostore/4/children?
   page=1&amp;pageSize=50" rel="last"></link>
     <entry>
       <title type="text">Alert Notifications</title>
       <id>tag:sap.com,2010:bip-rs/ARZB.BFCQk9PqaqDpcFwo1w</id>
       <author><name>System Account</name></author>
       <link href="http://localhost:6405/biprws/infostore/Alert%20Notifications"
   rel="alternate"></link>
       <content type="application/xml">
         <attrs xmlns="http://www.sap.com/rws/bip">
           <attr name="id" type="int32">64</attr>
           <attr name="cuid" type="string">ARZB.BFCQk9PqaqDpcFwo1w</attr>
           <attr name="description" type="string" null="true"></attr>
           <attr name="type" type="string">Folder</attr>
         </attrs>
       </content>
     </entry>
     <entry>
       <title type="text">Application Folder</title>
       <id>tag:sap.com,2010:bip-rs/AdoctK9h1sBHp3I6uG0Sh7M</id>
       <author><name>System Account</name></author>
       <link href="http://localhost:6405/biprws/infostore/Application%20Folder"
   rel="alternate"></link>
       <content type="application/xml">
         <attrs xmlns="http://www.sap.com/rws/bip">
           <attr name="id" type="int32">43</attr>
           <attr name="cuid" type="string">AdoctK9h1sBHp3I6uG0Sh7M</attr>
           <attr name="description" type="string"></attr>
           <attr name="type" type="string">Folder</attr>
         </attrs>
       </content>
     </entry>
   ...
   ```

```
</feed>
```

Related Information

Authentication [page 38]

4.5.2 To retrieve an object by ID

Before you can retrieve a resource from the BI platform, you must have a valid logon token and know the base URL for RESTful web service requests. To retrieve an object by ID, you must know the ID of the resource you are requesting. You can find the ID of a resource by accessing it in the Central Management Console (CMC) and inspecting its properties, or by reading the `id` attribute of the `<attr>` entry returned in a RESTful web service response. The ID attribute corresponds to the `SI_ID` property of the object in the BI platform repository.

You can access a resource directly by using its ID.

1. Create a new HTTP request.
2. Add the `X-SAP-LogonToken` attribute to the request header and set its value to a valid logon token.
3. Add the `Accept` attribute to the request header and set its value to `application/xml`.
4. Use the `GET` method to send a request to the `http://<baseURL>/biprws/infostore/<ID>` URL.

   Replace `<baseURL>` with the base URL for RESTful web service requests, and replace `<ID>` with the ID of the object you want to retrieve.

   ```
   GET http://localhost:6405/biprws/infostore/43
   ```

   The response contains an `<entry>` element that contains an XML description of the resource. This example gets the Application Folder by its ID, 43.

   ```
   <entry xmlns="http://www.w3.org/2005/Atom">
     <author><name>System Account</name></author>
     <id>tag:sap.com,2010:bip-rs/AdoctK9h1sBHp3I6uG0Sh7M</id>
     <title type="text">Application Folder</title>
     <updated>2011-04-14T10:27:50.672Z</updated>
     <link href="http://localhost:6405/biprws/infostore/Application%20Folder/
   children" rel="http://www.sap.com/rws/bip#children"></link>
     <link href="http://localhost:6405/biprws/infostore" rel="up"></link>
     <content type="application/xml">
       <attrs xmlns="http://www.sap.com/rws/bip">
         <attr name="id" type="int32">43</attr>
         <attr name="cuid" type="string">AdoctK9h1sBHp3I6uG0Sh7M</attr>
         <attr name="description" type="string"></attr>
         <attr name="name" type="string">Application Folder</attr>
         <attr name="type" type="string">Folder</attr>
       </attrs>
     </content>
   </entry>
   ```

4.5.3 To retrieve an object by CUID

Before you can retrieve a resource from the BI platform, you must have a valid logon token and know the base URL for RESTful web service requests. To retrieve an object by CUID, you must know the CUID of the resource you are requesting. You can find the CUID of a resource by accessing it in the Central Management Console (CMC) and inspecting its properties, or by reading the `cuid` attribute of the `<attr>` entry returned in a RESTful web service response. The CUID attribute corresponds to the `SI_CUID` property of the object in the BI platform repository.

You can access a resource directly by using its CUID.

1. Create a new HTTP request.
2. Add the `X-SAP-LogonToken` attribute to the request header and set its value to a valid logon token.
3. Add the `Accept` attribute to the request header and set its value to `application/xml`.
4. Use the `GET` method to send a request to `http://<baseURL>/infostore/cuid_<CUID>`.

   Replace `<baseURL>` with the base URL for RESTful web service requests, and replace `<CUID>` with the CUID of the object you want to retrieve. This example gets the Application Folder by its CUID, `AdoctK9h1sBHp3I6uG0Sh7M`.

   ```
   GET http://localhost:6405/biprws/infostore/cuid_AdoctK9h1sBHp3I6uG0Sh7M
   ```

   The response is an `<entry>` element that contains an XML description of the resource. In this example, the object with CUID = `AdoctK9h1sBHp3I6uG0Sh7M` is returned.

   ```
   <entry xmlns="http://www.w3.org/2005/Atom">
     <author><name>System Account</name></author>
     <id>tag:sap.com,2010:bip-rs/AdoctK9h1sBHp3I6uG0Sh7M</id>
     <title type="text">Application Folder</title>
     <updated>2011-04-14T10:27:50.672Z</updated>
     <link href="http://localhost:6405/biprws/infostore/Application%20Folder/
   children" rel="http://www.sap.com/rws/bip#children"></link>
     <link href="http://localhost:6405/biprws/infostore" rel="up"></link>
     <content type="application/xml">
       <attrs xmlns="http://www.sap.com/rws/bip">
         <attr name="id" type="int32">43</attr>
         <attr name="cuid" type="string">AdoctK9h1sBHp3I6uG0Sh7M</attr>
         <attr name="description" type="string"></attr>
         <attr name="name" type="string">Application Folder</attr>
         <attr name="type" type="string">Folder</attr>
       </attrs>
     </content>
   </entry>
   ```

4.5.4 To access child objects

Before you can retrieve a resource from the BI platform, you must have a valid logon token and know the base URL for RESTful web service requests.

You can access the children of a parent resource by appending `/children` to the end of the RESTful web service request.

1. Create a new HTTP request.
2. Add the `X-SAP-LogonToken` attribute to the request header and set its value to a valid logon token.

3. Add the `Accept` attribute to the request header and set its value to `application/xml`.
4. Use the `GET` method to send a request to the `http://<baseURL>/biprws/infostore/<id>/children` URL.

   Replace `<baseURL>` with the base URL for RESTful web service requests, and replace `<id>` with the ID or cuid_CUID of the parent object you want to retrieve.

   This example requests the children of the Root Folder by its ID, 23.

```
http://<baseURL>/biprws/infostore/23/children
```

   The response contains a `<feed>` element contains `<entry>` elements for each child of the requested resource. In this example, the children of the Root Folder are returned, including entries for Data Federation, Feature Samples, Web Intelligence Samples, and more.

```
<feed xmlns="http://www.w3.org/2005/Atom">
  <id>tag:sap.com,2010:bip-rs/ASHnC0S_Pw5LhKFbZ.iA_j4/children</id>
  <title type="text">Children of Root Folder</title>
  <updated>2011-04-15T00:31:16.609Z</updated>
  <link href="http://localhost:6405/biprws/infostore/Root%20Folder/children?
page=1&amp;pageSize=50" rel="self"></link>
  <link href="http://localhost:6405/biprws/infostore/Root%20Folder/children?
page=1&amp;pageSize=50" rel="first"></link>
  <link href="http://localhost:6405/biprws/infostore/Root%20Folder/children?
page=1&amp;pageSize=50" rel="last"></link>
  <entry>
    <title type="text">Data Federation</title>
    <id>tag:sap.com,2010:bip-rs/FnKsrkkctAcA8BAAALB7kkQAADAFzVMX</id>
    <author><name>System Account</name></author>
    <link href="http://localhost:6405/biprws/infostore/4044" rel="alternate"></
link>
    <content type="application/xml">
      <attrs xmlns="http://www.sap.com/rws/bip">
        <attr name="id" type="int32">4044</attr>
        <attr name="cuid" type="string">FnKsrkkctAcA8BAAALB7kkQAADAFzVMX</attr>
        <attr name="description" type="string" null="true"></attr>
        <attr name="type" type="string">Folder</attr>
      </attrs>
    </content>
  </entry>
  <entry>
    <title type="text">Feature Samples</title>
    <id>tag:sap.com,2010:bip-rs/AfoyR1BSRYJIgOkbmWfd3zU</id>
    <author><name>Administrator</name>
    <uri>http://localhost:6405/biprws/infostore/12</uri></author>
    <link href="http://localhost:6405/biprws/infostore/5158" rel="alternate">
</link>
    <content type="application/xml">
      <attrs xmlns="http://www.sap.com/rws/bip">
      <attr name="id" type="int32">5158</attr>
      <attr name="cuid" type="string">AfoyR1BSRYJIgOkbmWfd3zU</attr>
      <attr name="description" type="string">Contains examples of new features</
attr>
      <attr name="type" type="string">Folder</attr>
      </attrs>
    </content>
  </entry>

...
  <entry>
    <title type="text">Web Intelligence Samples</title>
    <id>tag:sap.com,2010:bip-rs/AeN4lEu0h_tAtnPEjFYxwi8</id>
    <author><name>Administrator</name>
    <uri>http://localhost:6405/biprws/infostore/12</uri></author>
```

```
    <link href="http://localhost:6405/biprws/infostore/4946" rel="alternate"></
link>
    <content type="application/xml">
      <attrs xmlns="http://www.sap.com/rws/bip">
      <attr name="id" type="int32">4946</attr>
      <attr name="cuid" type="string">AeN4lEu0h_tAtnPEjFYxwi8</attr>
      <attr name="description" type="string"></attr>
      <attr name="type" type="string">Folder</attr>
      </attrs>
    </content>
  </entry>
</feed>
```

4.5.5 To use pagination with results

Before you can retrieve a resource from the BI platform, you must have a valid logon token and know the base URL for RESTful web service requests.

When a response contains a large number of entries, you can divide the entries into pages and view one page at a time. You can set the number of entries that appear on a page, and then request the page number that you want to view.

> **Note**
>
> If you do not explicitly set the pagination information, then results are returned according to the default page size, which is set by an administrator. The default value is 50 entries per page.

1. Create a new HTTP request.
2. Add the `X-SAP-LogonToken` attribute to the request header and set its value to a valid logon token.
3. Add the `Accept` attribute to the request header and set its value to `application/xml`.
4. Append `?page=<n>&pageSize=<m>` to the end of the URL that requests a feed that contains multiple entries.

   Replace `<n>` with the page number of the page you want to view. Replace `<m>` with the number of entries to display on each page.

   This example requests to return the children of object with ID=23. It requests the second page of results, where each page contains three entries.

   ```
   http://<baseURL>/biprws/infostore/23/children?page=2&pageSize=3
   ```
5. Use the `GET` method to send the request.

The response contains a list of entries for the requested page. It also returns a set of links that you can use to see the first, last, next, and previous pages. This example shows the second page, where each page contains three entries.

```
<feed xmlns="http://www.w3.org/2005/Atom">
<id>tag:sap.com,2010:bip-rs/ASHnC0S_Pw5LhKFbZ.iA_j4/children</id>
<title type="text">Children of Root Folder</title>
<updated>2011-04-07T23:50:17.983Z</updated>
<link href="http://localhost:6405/biprws/infostore/Root%20Folder/children?
page=2&amp;pageSize=3" rel="self"></link>
<link href="http://localhost:6405/biprws/infostore/Root%20Folder/children?
page=1&amp;pageSize=3" rel="first"></link>
<link href="http://localhost:6405/biprws/infostore/Root%20Folder/children?
page=3&amp;pageSize=3" rel="next"></link>
```

```
<link href="http://localhost:6405/biprws/infostore/Root%20Folder/children?
page=1&amp;pageSize=3" rel="previous"></link>
<link href="http://localhost:6405/biprws/infostore/Root%20Folder/children?
page=3&amp;pageSize=3" rel="last"></link>
<entry>
  <title type="text">Platform Search Scheduling</title>
  <id>tag:sap.com,2010:bip-rs/AfbVaQ1CdrNDkKlzAKEK3aI</id>
  <author><name>System Account</name></author>
  <link href="http://localhost:6405/biprws/infostore/4320" rel="alternate"></link>
  <content type="application/xml">
    <attrs xmlns="http://www.sap.com/rws/bip">
      <attr name="id" type="int32">4320</attr>
      <attr name="cuid" type="string">AfbVaQ1CdrNDkKlzAKEK3aI</attr>
      <attr name="description" type="string" null="true"></attr>
      <attr name="type" type="string">Folder</attr></attrs>
  </content>
</entry>
<entry>
  <title type="text">Probes</title>
  <id>tag:sap.com,2010:bip-rs/AYtU9ijcgpxFsbgLW0om5_U</id>
  <author><name>System Account</name></author>
  <link href="http://localhost:6405/biprws/infostore/4001" rel="alternate"></link>
  <content type="application/xml">
    <attrs xmlns="http://www.sap.com/rws/bip">
      <attr name="id" type="int32">4001</attr>
      <attr name="cuid" type="string">AYtU9ijcgpxFsbgLW0om5_U</attr>
      <attr name="description" type="string" null="true"></attr>
      <attr name="type" type="string">Folder</attr>
    </attrs>
  </content>
</entry>
<entry>
  <title type="text">Report Conversion Tool</title>
  <id>tag:sap.com,2010:bip-rs/AY9zJ8BgaF9OucZ2h2slcJM</id>
  <author><name>Administrator</name>
  <uri>http://localhost:6405/biprws/infostore/12</uri></author>
  <link href="http://localhost:6405/biprws/infostore/4082" rel="alternate"></link>
  <content type="application/xml">
    <attrs xmlns="http://www.sap.com/rws/bip">
      <attr name="id" type="int32">4082</attr>
      <attr name="cuid" type="string">AY9zJ8BgaF9OucZ2h2slcJM</attr>
      <attr name="description" type="string"></attr>
      <attr name="type" type="string">Folder</attr>
    </attrs>
  </content>
  </entry>
</feed>
```


Related Information

Sorting and Filtering [page 141]

To set the default number of entries displayed on each page [page 79]

4.5.6 To filter results by type

Before you can retrieve a resource from the BI platform, you must have a valid logon token and know the base URL for RESTful web service requests.

You can limit the type of results returned by a response by appending `?type=<type>` to the end of the RESTful web service request. Replace `<type>` with the type of results you want to see. The `<type>` value corresponds to the `SI_KIND` property of the object in the BI platform repository.

1. Create a new HTTP request.
2. Add the `X-SAP-LogonToken` attribute to the request header and set its value to a valid logon token.
3. Add the `Accept` attribute to the request header and set its value to `application/xml`.
4. Append `?type=<type>` to the end of a URL that requests a feed that contains multiple entries.

   Replace `<type>` with the type of result you want to be returned. This example requests to return the children of the folder with ID=99 that have the type InfoView.

   ```
   http://<baseURL>/biprws/infostore/99/children?type=InfoView
   ```

5. Use the `GET` method to send the request.

   ```
   GET http://<baseURL>/biprws/infostore/99/children?type=InfoView
   ```

   The response contains a `<feed>` element that contains `<entry>` elements for children of object `99` that are of type `InfoView`.

   ```
   <feed xmlns="http://www.w3.org/2005/Atom">
     <id>tag:sap.com,2010:bip-rs/AWItAeqx.FpBgqTpFH8LqwE/children</id>
     <title type="text">Children of Root Folder 99</title>
     <updated>2011-06-06T23:40:10.209Z</updated>
     <link href="http://localhost:6405/biprws/infostore/99/children?
   page=1&pageSize=50" rel="self"></link>
     <link href="http://localhost:6405/biprws/infostore/99/children?
   page=1&pageSize=50" rel="first"></link>
     <link href="http://localhost:6405/biprws/infostore/99/children?
   page=1&pageSize=50" rel="last"></link>
     <entry>
       <title type="text">BI launch pad</title>
       <id>tag:sap.com,2010:bip-rs/Ac7UIwmYafpFuhiiw6FRXLQ</id>
       <author><name>System Account</name></author>
       <link href="http://localhost:6405/biprws/infostore/474" rel="alternate"></
   link>
       <content type="application/xml">
         <attrs xmlns="http://www.sap.com/rws/bip">
           <attr name="id" type="string">474</attr>
           <attr name="cuid" type="string">Ac7UIwmYafpFuhiiw6FRXLQ</attr>
           <attr name="description" type="string" null="true"></attr>
           <attr name="type" type="string">InfoView</attr>
         </attrs>
       </content>
     </entry>
   </feed>
   ```

4.5.7 To access objects with relationships

Before you can retrieve a resource from the BI platform, you must have a valid logon token and know the base URL for RESTful web service requests.

You can access objects that are related to the currently listed object by appending its object ID, then append `/relationships` to the URL followed by the name of the relationship. Further, you can make more specific queries by adding the ID or CUID of an object. A relationship can be, for example, a resource such as an account named

Administrator that is associated with with other objects such as user groups, received alerts and subscribed events. Use of the `/infostore/<id>` API will return relationship information on the InfoObject with `<id>` if such associations exist. For more information on relationships, consult the Business Intelligence Platform Administator Guide.

1. Create a new HTTP request.
2. Add the `X-SAP-LogonToken` attribute to the request header and set its value to a valid logon token.
3. Add the `Accept` attribute to the request header and set its value to `application/xml`.
4. Use the `GET` method to send a request to the `http://<baseURL>/biprws/infostore/<id>/relationships/<id>` URL.

   Replace `<baseURL>` with the base URL for RESTful web service requests, and replace `<id>` with the `ID` or `CUID` of the object you want to retrieve.

   To illustrate relationships, the following example begins by using the `/infostore` API. This will reveal if an object with an ID of `12` has any relationships to other objects in the BI platform.

```
http://commandcom-lcm:6405/biprws/infostore/12
```

   The response shows that ID 12 is an `Administrator` object that has relationships that include include `subscribedEvents`, `userGroups` and `receivedAlerts`.

```
<?xml version="1.0" ?>
<entry xmlns="http://www.w3.org/2005/Atom">
  <author>
    <name>
      Administrator
    </name>
    <uri>
      http://commandcom-lcm:6405/biprws/infostore/12
    </uri>
  </author>
  <id>
    tag:sap.com,2010:bip-rs/AfRWaT5_131NlLLf5bRMLKY
  </id>
  <title type="text">
    Administrator
  </title>
  <updated>
    2012-01-04T20:03:20.085Z
  </updated>
  <link href="http://commandcom-lcm:6405/biprws/infostore/Users" rel="up"/>
  <link href="http://commandcom-lcm:6405/biprws/infostore/12/relationships/
subscribedEvents"
          rel="http://www.sap.com/rws/bip#subscribed-events" title="Subscribed
events"/>
  <link href="http://commandcom-lcm:6405/biprws/infostore/12/relationships/
userGroups"
          rel="http://www.sap.com/rws/bip#user-groups" title="User groups"/>
  <link href="http://commandcom-lcm:6405/biprws/infostore/12/relationships/
receivedAlerts"
          rel="http://www.sap.com/rws/bip#received-alerts" title="Received
alerts"/>
  <content type="application/xml">
    <attrs xmlns="http://www.sap.com/rws/bip">
      <attr name="id" type="int32">
        12
      </attr>
      <attr name="cuid" type="string">
        AfRWaT5_131NlLLf5bRMLKY
      </attr>
      <attr name="description" type="string">
```

```
        Administrator account
      </attr>
      <attr name="name" type="string">
        Administrator
      </attr>
      <attr name="type" type="string">
        User
      </attr>
      <attr name="emailAddress" type="string"/>
      <attr name="lastLogon" type="datetime">
        2012-01-04T20:03:20.085Z
      </attr>
      <attr name="fullName" type="string"/>
    </attrs>
  </content>
</entry>
```

The following code snippet uses the `.../relationship/users` link obtained from the previous example.

```
http://commandcom-lcm:6405/biprws/infostore/12/relationships/userGroups
```

Since a trailing ID number was not used, the response in the following code snippet lists 3 links that may be examined further. These are `../infostore/1`, `../infostore/2` and `../infostore/3`.

```
<?xml version="1.0" ?>
<feed xmlns="http://www.w3.org/2005/Atom">
  <author>
    <name>
      Administrator
    </name>
    <uri>
      http://commandcom-lcm:6405/biprws/infostore/12
    </uri>
  </author>
  <id>
    tag:sap.com,2010:bip-rs/AfRWaT5_131NlLLf5bRMLKY/relationships/userGroups
  </id>
  <title type="text">
    InfoObjects related to Administrator via userGroups
  </title>
  <updated>
    2012-01-04T20:08:32.441Z
  </updated>
  <entry>
    <title type="text">
      1
    </title>
    <id>
      tag:sap.com,2010:bip-rs/AfRWaT5_131NlLLf5bRMLKY/relationships/userGroups/1
    </id>
    <link href="http://commandcom-lcm:6405/biprws/infostore/12/relationships/
userGroups/1" rel="self"/>
    <link href="http://commandcom-lcm:6405/biprws/infostore/1" rel="related"/>
    <content type="application/xml">
      <attrs xmlns="http://www.sap.com/rws/bip">
        <attr name="id" type="int32">
          1
        </attr>
      </attrs>
    </content>
  </entry>
  <entry>
    <title type="text">
      2
    </title>
    <id>
```

```
      tag:sap.com,2010:bip-rs/AfRWaT5_131NlLLf5bRMLKY/relationships/userGroups/2
    </id>
    <link href="http://commandcom-lcm:6405/biprws/infostore/12/relationships/
userGroups/2" rel="self"/>
    <link href="http://commandcom-lcm:6405/biprws/infostore/2" rel="related"/>
    <content type="application/xml">
      <attrs xmlns="http://www.sap.com/rws/bip">
        <attr name="id" type="int32">
          2
        </attr>
      </attrs>
    </content>
  </entry>
  <entry>
    <title type="text">
      3
    </title>
    <id>
      tag:sap.com,2010:bip-rs/AfRWaT5_131NlLLf5bRMLKY/relationships/userGroups/3
    </id>
    <link href="http://commandcom-lcm:6405/biprws/infostore/12/relationships/
userGroups/3" rel="self"/>
    <link href="http://commandcom-lcm:6405/biprws/infostore/3" rel="related"/>
    <content type="application/xml">
      <attrs xmlns="http://www.sap.com/rws/bip">
        <attr name="id" type="int32">
          3
        </attr>
      </attrs>
    </content>
  </entry>
</feed>
```

4.6 Scheduling

The Business Intelligence platform RESTful web service SDK enables you to perform basic scheduling.

> **Note**
>
> Only schedulable objects can be used with the scheduling API. Schedulable resources include documents, for example Crystal reports and Webl reports.

You can set the following scheduling properties:

- The time to schedule the resource.
- The recurrence properties of the resource, including the start time, end time, and recurrence interval.
  For example, a report could be scheduled to recur every Monday morning for the next year.
- The number of retries allowed and the retry interval.
  For example, if scheduling fails, you could allow up to three retries at hourly intervals.

4.6.1 To discover the scheduling URLs for an object

Before you can discover the URLs for scheduling an object, you must have a valid logon token and know the base URL for RESTful web service requests.

To get a list of URLs that can be used to schedule an object, append `/scheduleForms` to the end of a request for a schedulable resource. Schedulable resources include documents, for example Crystal reports and Webl reports.

1. Create a new HTTP request.
2. Add the `X-SAP-LogonToken` attribute to the request header and set its value to a valid logon token.
3. Add the `Accept` attribute to the request header and set its value to `application/xml`.
4. Use the `GET` method to send a request to the `http://<baseURL>/biprws/infostore/<id>/scheduleForms` URL.

   Replace `<baseURL>` with the base URL for RESTful web service requests, and replace `<id>` with the ID or CUID of a schedulable resource.

   ```
   GET http://localhost:6405/biprws/infostore/4738/scheduleForms
   ```

   The response contains a feed of entries that show the links for scheduling the resource. The following example shows links that you can use to schedule a report with the following recurrence:
   - now
   - once
   - hourly
   - daily
   - weekly
   - monthly
   - NthDayOfMonth

   > **Note**
   >
   > If the resource is not schedulable, an error is returned.

   ```
   <feed xmlns="http://www.w3.org/2005/Atom">
   <author>
     <name>Administrator</name>
     <uri>http://localhost:6405/biprws/infostore/12</uri>
   </author>
   <id>tag:sap.com,2010:bip-rs/ASb6ObslHktFnk3uF8.g3tw/scheduleForms</id>
   <title type="text">Schedule Drilldown</title><updated>2011-05-18T10:31:30.092Z</
   updated>
     <entry>
       <title type="text">now</title><id>tag:sap.com,2010:bip-rs/
   ASb6ObslHktFnk3uF8.g3tw/now</id>
       <link href="http://localhost:6405/biprws/infostore/5177/scheduleForms/now"
   rel="alternate"></link>
     </entry>
     <entry>
       <title type="text">once</title>
       <id>tag:sap.com,2010:bip-rs/ASb6ObslHktFnk3uF8.g3tw/once</id>
       <link href="http://localhost:6405/biprws/infostore/5177/scheduleForms/once"
   rel="alternate"></link>
     </entry>
     <entry>
       <title type="text">hourly</title>
       <id>tag:sap.com,2010:bip-rs/ASb6ObslHktFnk3uF8.g3tw/hourly</id>
   ```

```
    <link href="http://localhost:6405/biprws/infostore/5177/scheduleForms/
hourly" rel="alternate"></link>
  </entry>
  <entry>
    <title type="text">daily</title>
    <id>tag:sap.com,2010:bip-rs/ASb6ObslHktFnk3uF8.g3tw/daily</id>
    <link href="http://localhost:6405/biprws/infostore/5177/scheduleForms/daily"
rel="alternate"></link>
  </entry>
  <entry>
    <title type="text">weekly</title>
    <id>tag:sap.com,2010:bip-rs/ASb6ObslHktFnk3uF8.g3tw/weekly</id>
    <link href="http://localhost:6405/biprws/infostore/5177/scheduleForms/
weekly" rel="alternate"></link>
  </entry>
  <entry>
    <title type="text">monthly</title>
    <id>tag:sap.com,2010:bip-rs/ASb6ObslHktFnk3uF8.g3tw/monthly</id>
    <link href="http://localhost:6405/biprws/infostore/5177/scheduleForms/
monthly" rel="alternate"></link>
  </entry>
  <entry><title type="text">NthDayOfMonth</title>
    <id>tag:sap.com,2010:bip-rs/ASb6ObslHktFnk3uF8.g3tw/NthDayOfMonth</id>
    <link href="http://localhost:6405/biprws/infostore/5177/scheduleForms/
NthDayOfMonth" rel="alternate"></link>
  </entry>
</feed>
```

Related Information

Getting Template for Scheduling [page 103]

4.6.2 To schedule a resource

Before you can schedule a resource, you must have obtained a valid logon token and know the base URL for RESTful web service requests.

The Business Intelligence platform RESTful web service SDK allows for basic scheduling, including setting the time to schedule the resource and recurrence information.

> **Note**
>
> The scheduling APIs only work with objects that are schedulable. Schedulable resources include documents, for example Crystal reports and Webl reports.

1. Create a new HTTP request.
2. Add the `X-SAP-LogonToken` attribute to the request header and set its value to be a valid logon token.
3. Add the `Accept` attribute to the request header and set its value to `application/xml`.
4. Use the `GET` method to send a request to the `http://<baseURL>/biprws/infostore/<id>/scheduleForms/<form>` URL.

Replace `<baseURL>` with the base URL for RESTful web service requests, and replace `<ID>` with the ID or cuid_CUID of the resource. Replace `<form>` with the frequency of scheduling to perform, for example, `now`, `daily`, `weekly`, or `once`.

```
GET http://localhost:6405/biprws/infostore/4738/scheduleForms/now
```

The response contains an XML template that you can use to populate the request body of a request to schedule the resource.

```
<entry xmlns="http://www.w3.org/2005/Atom">
  <author>
    <name>Administrator</name>
    <uri>http://localhost:6405/biprws/infostore/12</uri>
  </author>
  <id>tag:sap.com,2010:bip-rs/ASb6ObslHktFnk3uF8.g3tw/scheduleForms/now</id>
  <title type="text">Schedule Drilldown now</title>
  <updated>2011-05-18T10:31:30.092Z</updated>
  <content type="application/xml">
    <attrs xmlns="http://www.sap.com/rws/bip">
      <attr name="retriesAllowed" type="int32">0</attr>
      <attr name="retryIntervalInSeconds" type="int32">1800</attr>
    </attrs>
  </content>
</entry>
```

5. Create a new HTTP request.
6. Add the `X-SAP-LogonToken` attribute to the request header and set its value to a valid logon token.
7. Add the `Accept` attribute to the request header and set its value to `application/xml`.
8. Add the `Content-Type` attribute to the request header and set its value to `application/xml`.
9. Fill out the template received from the `GET` request, and add it to the new request body.

   In this example, 3 retries are allowed in intervals of 1800 seconds.

```
<entry xmlns="http://www.w3.org/2005/Atom">
  <author>
    <name>Administrator</name>
    <uri>http://localhost:6405/biprws/infostore/12</uri>
  </author>
  <id>tag:sap.com,2010:bip-rs/ASb6ObslHktFnk3uF8.g3tw/scheduleForms/now</id>
  <title type="text">Schedule Drilldown now</title>
  <updated>2011-05-18T10:31:30.092Z</updated>
  <content type="application/xml">
    <attrs xmlns="http://www.sap.com/rws/bip">
      <attr name="retriesAllowed" type="int32">3</attr>
      <attr name="retryIntervalInSeconds" type="int32">1800</attr>
    </attrs>
  </content>
</entry>
```

10. Use the `POST` method to send the request to the scheduling URL.

```
POST http://localhost:6405/biprws/infostore/4738/scheduleForms/now
```

If the resource is scheduled successfully, the response header contains the status code `201 Created`, and provides a link to the location of the scheduled instance.

```
Location: http://localhost:6405/biprws/infostore/5619
```

Related Information

Creating Schedule Once [page 107]

4.7 Configuring RESTful Web Services for Tomcat Server

If you are using a non-WACS server such as Tomcat, you can set the RESTful Web service parameter values as part of server configuration.

To configure the RESTful web services for Apache Tomcat application server, follow the procedure given below:

1. Stop the Tomcat server.
2. Access the file `biprws.properties` in the following file-path: `<tomcat-location>\webapps\biprws\WEB-INF\config\custom`.
3. Specify the values of following parameters in the file based on your custom requirements and save the file.
   1. `<Default_Number_Of_Objects_On_One_Page>`
   2. `<Enterprise_Session_Token_Timeout_In_Minutes>`
   3. `<Session_Pool_Size>`
   4. `<Session_Pool_Timeout_In_Minutes>`
   5. `<Log_Level>`
   6. `<Log_Location>`
4. Restart the Tomcat server.

> **Note**
>
> The default version of `biprws.properties` is available at `<tomcat-location>\webapps\biprws\WEB-INF\config\default` and your configuration for above listed parameters in custom file path overrides the default settings once you restart the server.

4.8 Configuring RESTful Web Services for WebSphere Server

If you are using a non-WACS server such as WebSphere, you can set the RESTful Web service parameter values as part of server configuration.

To configure the RESTful web services for WebSphere application server, follow the procedure given below:

1. Stop the WebSphere server.
2. Access the file `biprws.properties` in the following file-path: `<WebSphere-location>\webapps\biprws\WEB-INF\config\custom`.
3. Specify the values of following parameters in the file based on your custom requirements and save the file.
   1. `<Default_Number_Of_Objects_On_One_Page>`
   2. `<Enterprise_Session_Token_Timeout_In_Minutes>`

3. `<Session_Pool_Size>`
4. `<Session_Pool_Timeout_In_Minutes>`
5. `<Log_Level>`
6. `<Log_Location>`

4. Restart the WebSphere server.

> **Note**
>
> The default version of `biprws.properties` is available at `<WebSphere-location>\webapps\biprws\WEB-INF\config\default` and your configuration for above listed parameters in custom file path overrides the default settings once you restart the server.

5 Administration and installation tasks

This section is about installing and configuring RESTful web services on a BI platform installation.

To perform the tasks in this section, you must be a BI platform administrator. Administrators can configure the RESTful web services environment, including setting default system values, enabling features, and enhancing performance settings.

The default installation of the BI platform includes RESTful web services. However, if you have performed a custom installation of the BI platform and did not include RESTful web services, you can install it separately. RESTful web services require an instance of the Web Application Container Server (WACS), and installing RESTful web services will install a copy of the WACS server if one does not already exist.

In a complex deployment environment, for example one that uses a proxy or multiple instances of the WACS server, you may need to configure the server name and port that is used to listen to RESTful web service requests.

> **Note**
>
> For additional information on complex deployment scenarios, see the "Managing Web Application Container Servers (WACS)" section of the *SAP BusinessObjects Business Intelligence Platform Administrator Guide*.

5.1 To install RESTful web services on Windows

You can use the Windows installer to add RESTful web services to your custom BI platform deployment. RESTful web services requires an instance of the Web Application Container Server (WACS), which is installed with RESTful web services if it does not already exist. RESTful web services was introduced in BI platform 4.0 to Feature Pack 3.

- If your BI platform 4.0 FP3 is a new installation, RESTful Web Services is automatically included in the installation. If you choose custom install, RESTful Web Services is selected in the feature tree by default.
- If you are upgrading from 4.0 SP2 to 4.0 FP3, after completing the upgrade, use the *Programs and Features Windows Control Panel*, *Uninstall/Change* feature to add the RESTful web service.

For more information about installing the BI platform on Windows, see the *Business Intelligence Platform Installation Guide for Windows*, section 5.8.1 To modify SAP BusinessObjects Business Intelligence platform.

1. Start the Business Intelligence platform Windows installer, and follow the installation instructions for a custom installation.
2. On the *Select Features* screen, expand *Instances* > *Servers* > *Platform Services* and select *RESTful Web Services*.
3. Continue the installation.
4. On the *HTTP Listening Port Configuration* dialog, enter the port number for listening to RESTful web service requests.

   By default, the port number is 6405.
5. Complete the installation.

If your BI platform installation uses a proxy or more than one WACS server, you may need to configure the base URL for RESTful web services.

Related Information

To configure the base URL for RESTful web services [page 78]

5.2 To install RESTful web services on Unix

You can use the Unix installer to add RESTful web services to your custom BI platform deployment. RESTful web services requires an instance of the Web Application Container Server (WACS), which is installed with RESTful web services if it does not already exist.

For more information about installing the BI platform on Unix, see the *Business Intelligence Platform Installation Guide for Unix*.

1. Start the Business Intelligence platform Unix installer, and follow the installation instructions for a custom installation.
2. On the *Select Features* dialog, expand *Instances* > *Servers* > *Platform Services* and select *RESTful Web Services*.
3. Continue with the installation.
4. On the *HTTP Listening Port Configuration* dialog, enter the port number for listening to RESTful web service requests.

   By default, this port is set to 6405.
5. Complete the installation.

If your BI platform installation uses a proxy or more than one WACS server, you may need to configure the base URL for RESTful web services.

Related Information

To configure the base URL for RESTful web services [page 78]

5.3 To configure web.xml to enable WinAD SSO

Configuring the RESTful web services to recognize Windows Active Directory Single Sign-On (WinAD SSO) requires edits to the `web.xml` configuration file, located on the BI platform server. For more information, see "Using the SDK > Authentication > To get a logon token using an Active Directory Single Sign-On (AD SSO) account" in the *Business Intelligence Platform RESTful Web Service Developer Guide*.

To have a client computer WinAD SSO login credentials recognized by the BI platform server, you must uncomment the `Kerberos Proxy filter` section of the `web.xml` and update values for `idm.realm`, `idm.princ` and `idm.keytab` that reflect the active directory environment used.

1. Locate the `web.xml` configuration at `<boe root>\SAP BusinessObjects Enterprise XI 4.0\java\pjs\services\RestWebService\biprws\WEB-INF\`. The following filepath is an example.

```
C:\Program Files (x86)\SAP BusinessObjects\SAP BusinessObjects Enterprise XI
4.0\java\
pjs\services\RestWebService\biprws\WEB-INF\web.xml
```

2. In the `web.xml` file, uncomment the Kerberos Proxy Filter section by adding a comment close tag `-->` before the `<filter>` tag, and remove the closing comment tag `-->`

```
  <!-- Kerberos Proxy Filter
    - Uncomment this filter and the corresponding filter-mapping to enable
Kerberos SSO
    - for Windows AD (secWinAD) authentication.
    - The following options must be specified (the rest are optional):
    -   idm.realm
    -   idm.princ
    -   idm.keytab (unless using password, see below)
  -->
  <filter>
    <filter-name>WrappedResponseAuthFilter</filter-name>
      .
      .
      .
  </filter>
  <filter-mapping>
    <filter-name>WrappedResponseAuthFilter</filter-name>
    <url-pattern>/logon/adsso</url-pattern>
  </filter-mapping>

</web-app>
```

3. Update the `<param-value>` for each setting of `idm.realm`, `idm.princ` and `idm.keytab` with those used in your active directory environment.

```
    <init-param>
      <param-name>idm.realm</param-name>
      <param-value>ADDOM.COM</param-value>
      <description>
          Required: Set this value to the Kerberos realm to use.
      </description>
    </init-param>
    <init-param>
      <param-name>idm.princ</param-name>
      <param-value>BOE120SIAVMBOESRVR/bo.service.addom.com</param-value>
      <description>
          Set this value to the Kerberos service principal to use.
          This will be a name of the form HTTP/fully-qualified-host.
          For example, HTTP/example.vintela.com
          If not set, defaults to the server's hostname and the
          idm.realm  property above.
      </description>
    </init-param>
    <init-param>
      <param-name>idm.kdc</param-name>
      <param-value></param-value>
      <description>
          The KDC against which secondary credentials must be validated
          This can be used for BASIC fallback or credential delegation.
          By default the KDC will be discovered automatically and this
```

```
        parameter must only be used if automatic discovery fails, or
        if a different KDC to the one discovered must automatically be used.
    </description>
  </init-param>
  <init-param>
    <param-name>idm.keytab</param-name>
    <param-value>C:/winnt/BOE120SIAVMBOESRVR.keytab</param-value>
    <description>
        The file containing the keytab that Kerberos will use for
        user-to-service authentication. If unspecified, SSO will default
        to using an in-memory keytab with a password specified in the
        com.wedgetail.idm.sso.password environment variable.
    </description>
  </init-param>
```

Note

The `idm.keytab` value refers to a filepath on the BI platform server. Values for `idm.realm` and `idm.prince` may be viewed from the Central Management Console. On the *Authentication* tab In the CMC, double-click *Windows AD*. The value for `idm.realm` is set with the *Default AD Domain* parameter, under *AD Configuration Summary*. The value for `idm.prince` is set with the *Service principal name* parameter, under *Authentication Options*.

4. Restart the WACS service so that the changes made to `web.xml` are recognized.
5. Use a client machien to verify that an AD SSO login token may be retrieved using the RESTful Web Services API, (for example, `http://<boe host>:6405/biprws/logon/adsso`).
6. Test the token by using a `GET` query including `X-SAP-LogonToken` in the header and using the `/infostore` API.

5.4 To configure Methods and Headers command line parameters

As an administrator, you can restrict what methods and headers may be used by RESTful web services, by adding the appropriate options to *Command Line Parameters* in the properties of your Web Application Container Service (WACS). Changes to the parameters require restarting the WACS service.

1. Log on to the Central Management Console as an administrator user.
2. Click *Servers*, and then click *Servers List*.
3. Right-click on your Web Application Container Server (WACS); for example, `MySIA.WebApplicationContainerServer`, and click *Properties*.
   The *Properties* tab for the WACS server appears.
4. In the *Command Line Parameters* area, enter the methods and headers that will be allowed.

   Each option group is enclosed by double quotes. Use Methods other than `GET`, `HEAD` and `POST`. Use commas to separate the option values such as `PUT` and `DELETE` as shown in the following example.

```
"-Dcom.sap.bip.rs.cors.extra.methods= PUT, DELETE"
"-Dcom.sap.bip.rs.cors.extra.headers= X-SAP-LogonToken, X-SAP-PVL, WWW-
Authenticate"
```

> **Note**
>
> The default value to allow all methods and headers is `*` (asterisk). Omitting the command line parameters entirely, has the same effect.

5. Click *Save and Close*.
6. Restart the service by right-clicking on the WACS server name, for example `MySIA.WebApplicationContainerServer` and click *Restart Server*.

5.5 To configure the base URL for RESTful web services

If your BI platform deployment uses a proxy server or contains more than one instance of the Web Application Container Server (WACS), you may need to configure the base URL for use with RESTful web services. Before you configure the base URL, you must know the server name and port number that listens to RESTful web service requests.

The base URL is used as part of every RESTful web service request. Developers programmatically discover the base URL and use it to direct RESTful web service requests to the correct server and port. The base URL is also used in RESTful web service responses to define hyperlinks to other RESTful resources.

> **Note**
>
> In default installations of the BI platform, the base URL is defined as `http://<servername>:6405/biprws`. Replace `<servername>` with the name of the server that hosts RESTful web services.

1. Log on to the Central Management Console (CMC) as an administrator.
2. In the CMC, click *Applications*.
   A list of applications is displayed.
3. Right-click *RESTful Web Service* > *Properties*.
   The *Properties* dialog box appears.
4. In the *Access URL* text box, type the name of the base URL for RESTful web services.
   For example, type `http://<servername>:<portnumber>/biprws`. Replace `<servername>` and `<portnumber>` with the name of the server and the port that listens to RESTful web service requests.

   > **Caution**
   >
   > - **Tomcat server, WACS server, and WebSphere server are supported** for RESTful web services APIs.
   > - The *Access URL* displays the WACS URL by **default**. If you wish to use the RESTful web services APIs on the Tomcat Web Server, ensure to modify the required`<server>` and `<port>` values in accordance.
   > - In SP06, **only Apache Tomcat server and WebSphere server are certified** for use with the RESTful web services (RWS) APIs.

5. Click *Save and Close*.

5.6 To enable the error message stack

As an administrator, you can configure the error messages returned by RESTful web services to include the error stack. The error stack provides extra debugging information that can be used to discover where errors have occurred.

> **Note**
>
> You may not want to enable the error stack in production scenarios, because it could provide information about the BI platform that you do not want to reveal to end users. It is recommended to enable the error stack in production scenarios as required for debugging, and to turn it off when it is no longer needed.

1. Log on to the Central Management Console as administrator user.
2. Click *Servers*, and then click *Servers List*.
3. Right-click on your Web Application Container Server (WACS); for example, right-click on `MySIA.WebApplicationContainerServer`, and click *Properties*.
   The *Properties* tab for the WACS server appears.
4. In the *RESTful Web Service* area, select *Show Error Stack*.
5. Click *Save and Close*.

Error stack information is included in RESTful web service error messages.

5.7 To set the default number of entries displayed on each page

When a RESTful web service response contains a feed with a large number of entries, the response can be divided into pages. You can configure the default number of entries that are displayed on each page. When developers make RESTful web service requests, they can specify the number of entries to display on each page. However, if they do not specify this value then the default page size is used.

1. Log on to the Central Management Console as an administrator.
2. Click *Servers*, and then click *Servers List*.
3. Right-click on your Web Application Container Server (WACS); for example, right click on `MySIA.WebApplicationContainerServer`, and click *Properties*.
   The *Properties* tab for the WACS server appears.
4. In the *RESTful Web Service* area, type the default page size in the *Default Number of Objects on One Page* text area.
5. Click *Save and Close*.

5.8 To set the timeout value of a logon token

Logon tokens expire after they have not been used for a certain amount of time. You can set the amount of time that an unused logon token remains valid.

> **Note**
>
> By default, the logon token timeout value is one hour.

1. Log on to the Central Management Console as an administrator.
2. Click *Servers*, and then click *Servers List*.
3. Right-click on your Web Application Container Server (WACS); for example, right click on `MySIA.WebApplicationContainerServer`, and click *Properties*.
   The *Properties* tab for the WACS server appears.
4. In the *RESTful Web Service* area, type the number of minutes for a logon token to be valid in the *Enterprise Session Token Timeout (minutes)* text area.
5. Click *Save and Close*.

5.9 To configure session pool settings

You can improve server performance by using a session pool. The session pool caches active RESTful web service sessions so they can be reused when a user sends another request that uses the same logon token in the HTTP request header. The session pool size defines the number of cached sessions to be stored at one time, and the session timeout value controls the amount of time that a session is cached.

You can set the session pool size and the session timeout value:

1. Log on to the Central Management Console (CMC) as an administrator.
2. Click *Servers*, and then click *Servers List*.
3. Right-click on your Web Application Container Server (WACS); for example, right-click on `MySIA.WebApplicationContainerServer`, and click *Properties*.
   The *Properties* tab for the WACS server appears.
4. Type the maximum number of sessions to cache in the *Session Pool Size* text box of the *RESTful Web Service* area.
5. Type the session pool timeout value in the *Session Pool Timeout (minutes)* text box of the *RESTful Web Service* area.
6. Click *Save and Close*.
7. Right-click on the WACS server, for example, `MySIA.WebApplicationContainerServer`, and click *Restart Server*.

5.10 To enable HTTP basic authentication

HTTP basic authentication lets users make RESTful web service requests without providing a logon token. If HTTP basic authentication is enabled, users are prompted to provide their user name and password the first time they make a RESTful web service request.

> **Note**
>
> User names and passwords are not transmitted securely with HTTP basic authentication, unless it is used in conjunction with HTTPS.

When you enable HTTP basic authentication, you set the default HTTP basic authentication type to SAP, Enterprise, LDAP, or WinAD. Users can override the default HTTP basic authentication type when they log on.

Logging on to the BI platform using HTTP basic authentication consumes a license. If the session pool caching is used, the request uses the license associated with its cached session. If session pool caching is not used, a license is consumed while the request is in progress and released once the request is finished.

1. Log on to the Central Management Console (CMC) as an administrator.
2. Click *Server* > *Servers List*.
3. Right-click on your Web Application Container Server (WACS); for example, right-click on `MySIA.WebApplicationContainerServer`, and click *Properties*.
   The *Properties* tab for the WACS server appears.
4. In the *RESTful Web Service* area, select *Enable HTTP Basic Authentication*.
5. (Optional) In the *Default Authentication Scheme for HTTP Basic* list, select the default type of HTTP basic authentication.
6. Click *Save and Close*.

When an end user logs on using HTTP basic authentication, they can specify the type of authentication to use. In a web browser, the user types `<authtype>\<username>` in the user name prompt, and `<password>` in the password prompt.

To log on using HTTP basic authentication programmatically, users add the `Authorization` attribute to the HTTP request header, and set the value to be `Basic <authtype>\<username>:<password>`.

Replace `<authtype>` with the authentication type, `<username>` with the user name, and `<password>` with the password. The authentication type, user name, and password must be base64-encoded as defined by RFC 2617. User names that contain the `:` character cannot be used with HTTP basic authentication.

Related Information

To configure session pool settings [page 80]

5.11 To configure cross-origin resource sharing (CORS)

The *Cross-Origin Resource Sharing Configuration* (CORS) setting allows you to add a list of domain names to let users retrieve data from multiple sources on JavaScript-based web pages. This is necessary to get around the security policy that JavaScript and Ajax languages employ to prevent cross-domain access. To avoid compromising security, only those websites that may be accessed are added to the *Allow Origins* WACS server properties in CMC.

A *Max Age (minutes)* setting is also available to adjust the cache expiry time, which sets the maximum number of minutes that browsers can retain HTTP requests.

> **Note**
>
> By default, access to any and all domains are allowed with * (asterisk).

1. Log on to the Central Management Console as an administrator.
2. Click *Server* > *Servers List*.
3. Right-click on your Web Application Container Server (WACS); for example, right-click `MySIA.WebApplicationContainerServer`, and click *Properties*.
   The *Properties* tab for the WACS server appears.
4. In the *RESTful Web Service* area, go to the *Cross-Origin Resource Sharing Configuration* text box beside *Allow Origins:* and replace the * (asterisk) with your list of domain names, each separated by a comma. For example: `http://origin1.server:8080, http://origin2.server:8080`
5. In the *Max Age (minutes):* text box, type the maximum number of minutes that you want browsers to cache HTTP requests.
6. Click *Save and Close*.

5.12 To enable and configure trusted authentication

Trusted authentication is activated and configured through the Central Management Console (CMC) in areas that include *Authentication > Enterprise*, where Trusted Authentication is enabled and a shared secret key file is generated; *Users and Groups > User List*, where an account is created for a trusted user; and *Servers > Servers List > WACS > Properties*, where the *Retrieving Method* option is selected for `/logon/trusted` API logon token requests.

1. Log on to the Central Management Console as an administrator.
2. Go to *Authentication* > *Enterprise*, and then click *Trusted Authentication is enabled*.
3. Click *New Shared Secret*, and click *Download Shared Secret*.
4. Click *Save* and place the `TrustedPrincipal.conf` file in the default location, which is `<EnterpriseDir>\<platform>`.
   An example location appears as follows:

   ```
   "C:\Program Files (x86)\SAP BusinessObjects\SAP BusinessObjectsEnterprise XI 4.0\win64_x64\"
   ```

> **Note**
>
> You can change the default location of the `TrustedPrincipal.conf` shared secret file by adding a command line entry in the CMC at *Servers > Servers List > WACS > Properties > Command Line Parameters*, and then restarting the WACS service. For example, a command line entry using `-Dbobj.trustedauth.home=` and the folder `SharedSecrets` placed at the root of the `C:\` drive of the BI platform server would appear as follows:
>
> ```
> "-Dbobj.trustedauth.home=C:\SharedSecrets"
> ```

> **Note**
>
> You can leave the option *Shared Secret Validity Period (days)* at the default value of zero (0) so that it does not expire. The *Trusted logon request is timeout after N millisecond(s) (0 means no limit)* option can be left at the default value of zero (0) so that there is no time limit for trusted logon requests.

5. Click *Update* to save the change.
6. Add a new user and password, for example `bob` and `Passw0rd`, in *Users and Groups* > *User List* using *Manage > New > New user*. Uncheck *User must change password at next logon*, then click *Create & Close*.

> **Note**
>
> You can also create a new user by clicking the *Create new user* icon, or by right-clicking in an open area of the window that lists user names, and select *New* > *New User*.

7. Go to *Servers* > *Core Services* > *WACS* > *Properties*, scroll down to the *Trusted Authentication Configuration* section and use the *Retrieving Method* menu to select either *HTTP_HEADER*, *QUERY_STRING* or *COOKIE*.

> **Note**
>
> You can optionally change the *User Name Parameter* from the default label of `X-SAP-TRUSTED-USER` to any other convenient label, (for example `UserName`, `bankteller`, or `nurse`) that RESTful web services developers must use.

8. Restart the service by right-clicking on the WACS server name, for example `MySIA.WebApplicationContainerServer`, and click *Restart Server*.

> **Note**
>
> Later changing the option under *Retrieving Method* as shown in step 7 does not require restarting WACS.

9. Verify that you are able to retrieve a logon token by using the `.../biprsw/logon/trusted/` API and sending a `GET` request with the default header label of `X-SAP-TRUSTED-USER` with the user name created in step 6.

5.13 Securing Microsoft Silverlight access to the WACS server

Microsoft Silverlight components that are hosted in external applications can access the BI platform by using the Business Intelligence platform RESTful web service SDK. As an administrator, you can enhance the security of the

BI platform by restricting which domains are authorized to make Silverlight requests to applications hosted by the Web Application Container Server (WACS), including RESTful web services.

> **Note**
>
> Default installations of the BI platform allow unrestricted access to the WACS server by external Silverlight components.

The Silverlight access policy is defined by the `ClientAccessPolicy.xml` file, which is shared by all instances of the WACS server in a BI platform installation. Modifying the Silverlight access policy file changes the Silverlight access restrictions for all applications that are hosted by any WACS server in a BI platform deployment. This includes RESTful web services, and may include other BI platform web applications such as the Central Management Console (CMC) and BI Launch Pad if they are hosted by the WACS server.

> **Note**
>
> RESTful web services are always hosted by a WACS server and cannot be hosted by another type of servlet container.

The `ClientAccessPolicy.xml` file is located at `$ENTERPRISEDIR/warfiles/webapps/ROOT`, where `$ENTERPRISEDIR` represents the location of your BI platform installation. Modify this file to change the Silverlight policy settings. After editing this file, you must restart the WACS servers for the changes to take effect.

For more information about how to edit a Silverlight policy file, consult the Microsoft Silverlight product documentation.

6 Configuring Recipient Delivery Rules

7 API Reference

7.1 About API Reference and Prerequisites

The API reference lists the methods (such as GET, POST, DELETE) and the URLs using which you access the BI platform with the RESTful web service SDK.

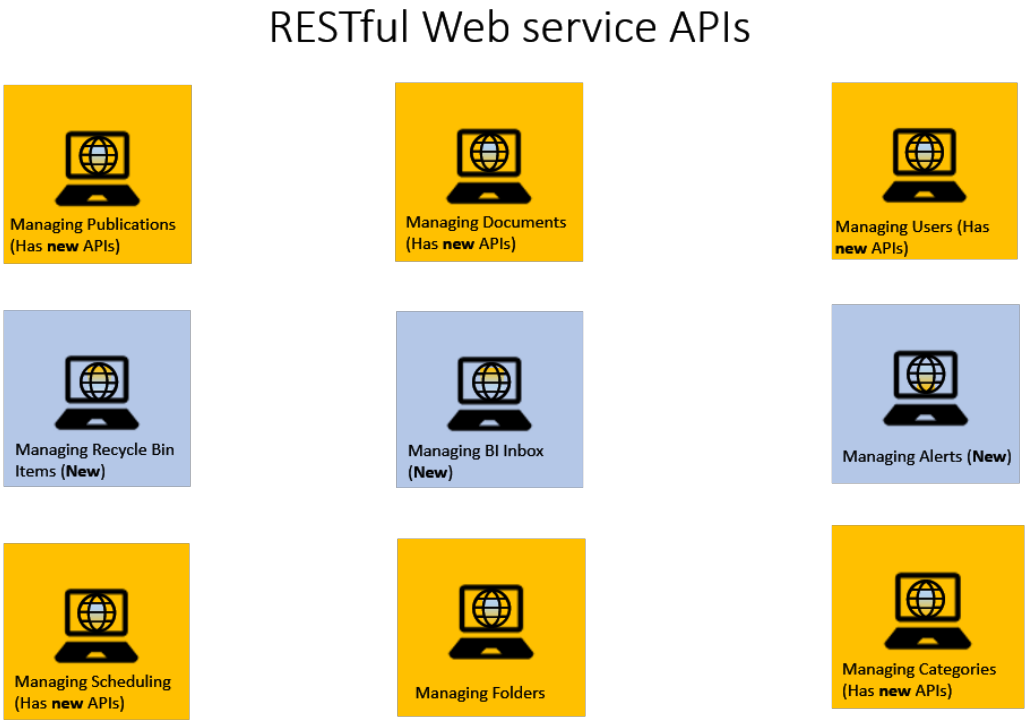
For the tasks listed in this section, it is assumed that you have:

- **A BI platform server machine running** (to which you log on using the BI launch pad).
- **A Web client (such as Postman) to interact with the server** using the URLs with respective methods.

> **Note**
>
> The <base URL> mentioned everywhere within URL syntax in the topics of *API Reference* is of the format `host:port/<root>` and is specific to your RESTful web server configuration. For example, the base URL can be `10.142.800.97:6405/biprws`.

Overview of Available APIs

The following image gives an overview of the RESTful Web service APIs available currently on the BI platform SDK (click on a service to navigate to the section detailing the APIs):



- Managing Users [page 146]
- Managing Documents [page 284]
- Managing Folders [page 273]
- Managing Alerts and Notifications [page 304]
- Managing Categories [page 262]
- Managing Recycle Bin Items [page 300]
- About API Reference and Prerequisites [page 86]
- Managing Publications [page 188]
- Managing User Groups [page 165]

Related Information

Retrieving the base URL for RESTful web service requests [page 15]

7.2 Fetching Information About the Build

The About Information API URL displays information about the build.

- Use the `GET` method to retrieve the information in an XML template.

GET http://<baseURL>/v1/about

Make a `GET` request to `/v1/about` to receive the information in a template

Request:

- Method: GET
- URL: `http://<baseURL>/v1/about`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`

> **Note**
>
> About information does not require any authorization that is not a value for X-SAP-LOGONTOKEN in the header

- Body: none

Response:

- Body: An XML template with About information is displayed.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">About</title>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="copyright" type="string">©2010 - 2016 SAP SE or an SAP
affiliate company.  All rights reserved.</attr>
            <attr name="build" type="string">14.2.3.516</attr>
            <attr name="vendor" type="string">SAP SE</attr>
            <attr name="timestamp" type="string">20160224.122139</attr>
        </attrs>
    </content>
</entry>
```

7.3 Fetching the Timezone Information

Timezone API URL displays timezone information for the application server, which deploys the web service.

- Use the `GET` method to retrieve the information in an XML template.

GET http://<baseURL>/v1/timezone

Make a `GET` request to `/v1/timezone` to receive the information in a template

Request:

- Method: GET
- URL: `http://<baseURL>/v1/timezone`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks

- Body: none

Response:

- Body: An XML template with timezone information is displayed.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="displayname" type="string">Central European Time</attr>
            <attr name="name" type="string">CET</attr>
        </attrs>
    </content>
</entry>
```

7.4 Authentication

7.4.1 /v1/logon/long

Log on to the BI platform with a username and password by making requests to the `/logon/long` URL.

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to log on to the BI platform and obtain a logon token.

GET http://<baseURL>/v1/logon/long

Make a `GET` request to `/logon/long` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/logon/long`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Accept`	`application/xml`

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<attrs xmlns="http://www.sap.com/rws/bip">
    <attr name="clienttype" type="string"></attr>
    <attr name="password" type="string"></attr>
    <attr name="auth" type="string"
possibilities="secEnterprise,secLDAP,secWinAD,secSAPR3">secEnterprise</attr>
    <attr name="username" type="string"></attr>
</attrs>
```

POST http://<baseURL>/logon/long

To receive a logon token, make a `POST` request to `/logon/long`, providing your user name and password.

Request:

- Method: `POST`
- URL: `http://<baseURL>/logon/long`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`

- Body:

```
<attrs xmlns="http://www.sap.com/rws/bip">
    <attr name="clienttype" type="string"></attr>
    <attr name="password" type="string"></attr>
    <attr name="auth" type="string"
possibilities="secEnterprise,secLDAP,secWinAD,secSAPR3">secEnterprise</attr>
    <attr name="username" type="string"></attr>
</attrs>
```

  - Use `<attr name="clienttype" type="string"></attr>` to define client type identifier.
  - Use `<attr name="password" type="string"></attr>` to define the password.
  - Use `<attr name="auth" type="string"></attr>` to define the type of authentication. Use one of `secEnterprise`, `secLDAP`, `secWinAD`, or `secSAPR3`.
  - Use `<attr name="username" type="string"></attr>` to define the user name.

> **Note**
>
> If any BOE client (Analysis Office, Xcelsius, LiveOffice etc.) using RESTful web services wants to be audited by its `<clienttype>`, then in the request, while creating the login session, set the `<clienttype>` attribute. This particular `<clientType>` is saved in the auditing database's `<ADS_EVENT>` table under `<clientType>`column.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
X-SAP-LogonToken	A logon token.
Date	Date and time of the response
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

The `X-SAP-LogonToken` attribute contains the logon token.

```
X-SAP-LogonToken:"BI42SP03TF22:6400@{3&amp;2=6883,U3&amp;
2v=BI42SP03TF22:6400,UP&amp;66=60,U3&amp;
68=secEnterprise:Administrator,UP&amp;S9=12,U3&amp;qe=100,U3&amp;vz=4B.wHI5lWQ.SX
HD7fEbc91qRmDdxju4oKHNr0OBJBTQ,UP}"
```

The logon token is contained between the quotation marks. In the example above, the logon token is as follows:

```
BI42SP03TF22:6400@{3&amp;2=6883,U3&amp;2v=BI42SP03TF22:6400,UP&amp;66=60,U3&amp;
68=secEnterprise:Administrator,UP&amp;S9=12,U3&amp;qe=100,U3&amp;vz=4B.wHI5lWQ.SX
HD7fEbc91qRmDdxju4oKHNr0OBJBTQ,UP}
```

- Body:
  The response body contains a copy of the logon token in the `<attr name="logonToken" type="string">` element. The logon token must be converted from its XML-encoded format to its original format before it can be used. For example, replace the `&` character sequence with the `&` character.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>@BI42SP03TF22:6400</name>
    </author>
    <title type="text">Logon Result</title>
    <updated>2016-06-07T04:08:01.688Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="logontoken" type="string">BI42SP03TF22:6400@{3&amp;
2=6883,U3&amp;2v=BI42SP03TF22:6400,UP&amp;66=60,U3&amp;
68=secEnterprise:Administrator,UP&amp;S9=12,U3&amp;qe=100,U3&amp;vz=4B.wHI5lWQ.SX
HD7fEbc91qRmDdxju4oKHNr0OBJBTQ,UP}</attr>
        </attrs>
    </content>
</entry>
```

  This example shows the returned logon token in the response body:

```
BI42SP03TF22:6400@{3&amp;2=6883,U3&amp;2v=BI42SP03TF22:6400,UP&amp;66=60,U3&amp;
68=secEnterprise:Administrator,UP&amp;S9=12,U3&amp;qe=100,U3&amp;vz=4B.wHI5lWQ.SX
HD7fEbc91qRmDdxju4oKHNr0OBJBTQ,UP}
```

  To use this logon token, convert it to its original format:

```
BI42SP03TF22:6400@{3&;2=6883,U3&;2v=BI42SP03TF22:6400,UP&;66=60,U3&;
68=secEnterprise:Administrator,UP&;S9=12,U3&;qe=100,U3&;vz=4B.wHI5lWQ.SXHD7fEbc91
qRmDdxju4oKHNr0OBJBTQ,UP}
```

Related Information

Converting a logon token from XML-encoded text [page 46]
To get a logon token from a user name and password [page 39]

7.4.2 /v1/logon/token

Log on to the BI platform with a serialized session or session token obtained from an existing serialized session by making requests to the `/v1/logon/token` URL.

- Use the `GET` method to retrieve an XML template for the request body.

- Use the `POST` method to log on to the BI platform and obtain a logon token.

GET http://<baseURL>/v1/logon/token

Make a `GET` request to `/v1/logon/token` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/logon/token`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Accept`	`application/xml`

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<attrs xmlns="http://www.sap.com/rws/bip">
    <attr name="clienttype" type="string"></attr>
    <attr name="logontoken" type="string"></attr>
    <attr name="tokentype" type="string" possibilities="token,
serializedSession">token</attr>
</attrs>
```

POST http://<baseURL>/v1/logon/token

To receive a logon token, make a `POST` request to `/v1/logon/token`, providing a serialized session or session token obtained from another SDK.

Request:

- Method: `POST`

- URL: `http://<baseURL>/v1/logon/token`.
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`

- Body:

```
<attrs xmlns="http://www.sap.com/rws/bip">
    <attr name="clienttype" type="string"></attr>
    <attr name="logontoken" type="string"></attr>
    <attr name="tokentype" type="string" possibilities="token,
serializedSession">token</attr>
</attrs>
```

  - Use `<attr name="tokentype" type="string" possibilities="token, serializedSession">` to define the type of token.
    Use `token` if you are providing a session token. Use `serializedSession` if you are providing a serialized session.
  - Use `<attr name="logontoken" type="string">` to define the serialized session or session token value.

    Note

    The serialized session or session token value must be XML-encoded to remove illegal XML characters. For example, replace the `&` character with `&`.

Response:

- Header:

Attribute	Value
Status code	HTTP response code.
Server	Type of server.
X-SAP-LogonToken	A logon token.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

The `X-SAP-LogonToken` attribute contains the logon token. The logon token is contained between the quotation marks.

```
X-SAP-LogonToken:"BI42SP03TF22:6400@{3&amp;2=6883,U3&amp;
2v=BI42SP03TF22:6400,UP&amp;66=60,U3&amp;
68=secEnterprise:Administrator,UP&amp;S9=12,U3&amp;qe=100,U3&amp;vz=4B.wHI5lWQ.SX
HD7fEbc91qRmDdxju4oKHNr0OBJBTQ,UP}"
```

- Body:

The response body contains an XML-encoded copy of the logon token in the `<attr>` element. The logon token must be converted from its XML-encoded format to its original format. For example, replace the `&` character sequence with the `&` character.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>@BI42SP03TF22:6400</name>
    </author>
    <title type="text">Logon Result</title>
    <updated>2016-06-07T04:39:40.167Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="logontoken" type="string">BI42SP03TF22:6400@{3&amp;
2=6883,U3&amp;2v=BI42SP03TF22:6400,UP&amp;66=60,U3&amp;
68=secEnterprise:Administrator,UP&amp;S9=12,U3&amp;qe=100,U3&amp;vz=4B.wHI5lWQ.SX
HD7fEbc91qRmDdxju4oKHNr0OBJBTQ,UP}</attr>
        </attrs>
    </content>
</entry>
```

This example shows the returned logon token in the response body:

```
BI42SP03TF22:6400@{3&amp;2=6883,U3&amp;2v=BI42SP03TF22:6400,UP&amp;66=60,U3&amp;
68=secEnterprise:Administrator,UP&amp;S9=12,U3&amp;qe=100,U3&amp;vz=4B.wHI5lWQ.SX
HD7fEbc91qRmDdxju4oKHNr0OBJBTQ,UP}</attr>
```

To use this logon token, convert it to its original format:

```
BI42SP03TF22:6400@{3&;2=6883,U3&;2v=BI42SP03TF22:6400,UP&;66=60,U3&;
68=secEnterprise:Administrator,UP&;S9=12,U3&;qe=100,U3&;vz=4B.wHI5lWQ.SXHD7fEbc91
qRmDdxju4oKHNr0OBJBTQ,UP}
```

Related Information

Converting a logon token from XML-encoded text [page 46]
To get a logon token from a serialized session or session token [page 41]

7.4.3 /v1/logon/adsso

The `/v1/logon/adsso` (Active Directory Single Sign On - ADSSO) is used to acquire tokens from Active Directory user accounts. The BOE server must have `web.xml` configured for ADSSO and users' Windows Active Directory login name must match their BOE account name.

- Use the `GET` method to retrieve the logon token.

GET http://<baseURL>/v1/logon/adsso

Make a `GET` request to `/v1/logon/adsso` to receive a logon token.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/logon/adsso`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
  Query Parameter:`?clienttype=<clienttype>`

> **Note**
>
> If any BOE client (Analysis Office, Xcelsius, LiveOffice etc.) using RESTful web services wants to be audited by its `<Clienttype>`, then in the client, while creating the login session, set the `<ClientType>` attribute (as shown above in "Query parameter") . This particular `<Clienttype>` is saved in the auditing database's `<ADS_EVENT>` table under `<Client_type>`column.

- Header:

Name	Value	Note
`Accept`	`application/xml`	Used by default, so use of this header is not necessary.

- Request Body: None
- Response Header

Name	Value	Example
Status Code	HTTP response code.	200 OK
Server	Type of server.	Apache-Coyote/1.1
X-SAP-LogonToken	Returned encoded token.	COMMANDCOM-LCM: 6400@{3&2=588...9QO0XnE,UP}
Date	Date and time of response.	Fri, 16 Dec 2011 22:00:57 GMT
Content-Type	Type of content in the response body.	application/xml
Content-Length	Length of content in the response body.	6919

Response Body:

```
<?xml version="1.0" ?>
<entry xmlns="http://www.w3.org/2005/Atom">
  <author>
    <name>
      @VMBOESRVR.ADDOM.COM
    </name>
  </author>
  <id>
    tag:sap.com,2010:bip-rs/logon/adsso
  </id>
  <title type="text">
    Logon Result
  </title>
  <updated>
    2011-11-21T22:15:51.340Z
  </updated>
  <content type="application/xml">
    <attrs xmlns="http://www.sap.com/rws/bip">
      <attr name="logonToken" type="string">
```

```
        VMBOESRVR.ADDOM.COM:6400@{3&2=4584,U3&p=40868.9276775116,Y7&4F=4331,
        U3&63=secWinAD,0P&66=60,03&68=secWinAD:CN%3DADUser1%2CCN%3DUsers%2CDC
%3D2K8ADDOMAIN
        %2CDC%3DCOM,0P&qe=100,U3&vz=kOox8TDqAiFsfs8T3GefI3sWXIyKymc9qvytAjihC7w,UP}
      </attr>
    </attrs>
  </content>
</entry>
```

> **Note**
>
> Internet Explorer can be used to retrieve an Active Directory single sign on `logontoken` by entering `http://<baseURL>/logon/adsso`. However, the returned value includes <name>, <id> and <updated> strings, data that is not part of a valid `logontoken`. The following text clipping shows irrelevant data that is prefixed to a `logonToken` request obtained with Internet Explorer.
>
> ```
> @VMBOESRVR.ADDOM.COMtag:sap.com,2010:bip-rs/logon/adsso2011-11-21T19:02:00.761Z
> ```
>
> The following text clipping shows a valid `logontoken` without extraneous data.
>
> ```
> BI42SP03TF22:6400@{3&;2=6990,U3&;2v=BI42SP03TF22:6400,UP&;66=60,U3&;
> 68=secEnterprise:Administrator,UP&;S9=12,U3&;qe=100,U3&;vz=eh1rLdKtjx1bKRnyRppRzSG
> 2eYnGWjBBlUVd1BoQRIY,UP}
> ```

Related Information

Converting a logon token from XML-encoded text [page 46]
To configure web.xml to enable WinAD SSO [page 75]

7.4.4 /v1//logon/trusted

GET http://<baseURL>/v1/logon/trusted

Used to retrieve a logon token using a trusted authenticated user name by sending a `GET` request to `/logon/trusted` by one of three retrieval methods: an HTTP header request, an HTTP-encoded URL query, or a cookie.

The following retrieving methods are available in CMC: *Servers > Servers List > WACS > Trusted Authentication Configuration*.

Retrieving Method
HTTP_HEADER
QUERY_STRING
COOKIE

The *User Name* parameter can be changed in CMC: *Servers > Core Services > WACS > Users Name Parameter*.

User Name Paramater	String value restrictions
X-SAP-TRUSTED-USER	Default setting. Cannot contain spaces or a colon (:)

> **Note**
>
> The WACS service does not need a restart between changes to the *Retrieving Method* or *User Name Parameter*. Query String URLs must be HTTP encoded. In generally, characters such as spaces and colons must not be used within values or name parameters.

1. Request using HTTP_HEADER:

- Method: GET
- URL: http://<baseURL>/v1/logon/trusted
  Replace <baseURL> with the base URL for RESTful web service requests.
- Query Parameter:?clienttype=<clienttype>

  > **Note**
  >
  > If any BOE client (Analysis Office, Xcelsius, LiveOffice etc.) using RESTful web services wants to be audited by its <Clienttype>, then in the client, while creating the login session, set the <ClientType> attribute (as shown above in "Query parameter") . This particular <Clienttype> is saved in the auditing database's <ADS_EVENT> table under <Client_type>column.

- Header: none

2. Request using QUERY_STRING:

- Method: GET
- URL: http://<baseURL>/v1/logon/trusted?<X-SAP-TRUSTED-USER>=<trustedUserName>
  Replace <baseURL> with the base URL for RESTful web service requests. The default label is <X-SAP-TRUSTED-USER>. This label can be changed to another value in CMC: *Servers > WACS*, *Trusted Authentication Configuration*, *User Name Parameter*. Replace <trustedUserName> with the name of a trusted user account name as defined in *CMC > Users and Groups*.
- Query Parameter:?clienttype=<clienttype>

  > **Note**
  >
  > If any BOE client (Analysis Office, Xcelsius, LiveOffice etc.) using RESTful web services wants to be audited by its <Clienttype>, then in the client, while creating the login session, set the <ClientType> attribute (as shown above in "Query parameter") . This particular <Clienttype> is saved in the auditing database's <ADS_EVENT> table under <Client_type>column.

- Header: none
- Restriction: URL must be HTTP encoded

3. Request using COOKIE:

- Method: GET
- URL: http://<baseURL>/v1/logon/trusted
  Replace <baseURL> with the base URL for RESTful web service requests.

- Query Parameter:`?clienttype=<clienttype>`

> **Note**
>
> If any BOE client (Analysis Office, Xcelsius, LiveOffice etc.) using RESTful web services wants to be audited by its `<Clienttype>`, then in the client, while creating the login session, set the `<ClientType>` attribute (as shown above in "Query parameter") . This particular `<Clienttype>` is saved in the auditing database's `<ADS_EVENT>` table under `<Client_type>`column.

- Header: none
- Cookie values:

Cookie category	Example value	Note
`Domain`	www.sap.com	This value is used as the server address in `<server address>`
Name	`X-SAP-TRUSTED-USER`	This is the default label. It may be changed in *CMC > Servers > WACS*, *Trusted Authentication Configuration*, *User Name Parameter*
Value	`bob`	The name of the trusted user as defined in *CMC > Users and Groups*
Path	/	Path local to the <server address>. Usually this is a forward slash (/).

- Response Header:

Name	Value	Example
Status Code	HTTP response code.	200 OK
Server	Type of server.	Apache-Coyote/1.1
X-SAP-LogonToken	Returned encoded token.	COMMANDCOM-LCM:6400@{3&2=5613,U3&p=40884.81...RrzfQ,UP}
Date	Date and time of response.	Wed, 07 Dec 2011 19:29:49 GMT
Content-Type	Type of content in the response body.	text/html
Content-Length	Length of content in the response body.	577

Response Body example:

```
<?xml version="1.0" ?>
<entry xmlns="http://www.w3.org/2005/Atom">
  <author>
    <name>
      @COMMANDCOM-LCM:6400
    </name>
  </author>
  <id>
    tag:sap.com,2010:bip-rs/logon/trusted
  </id>
  <title type="text">
    Logon Result
  </title>
```

```
  <updated>
    2011-12-07T21:46:57.091Z
  </updated>
  <content type="application/xml">
    <attrs xmlns="http://www.sap.com/rws/bip">
      <attr name="logonToken" type="string">
        COMMANDCOM-LCM:
6400@{3&2=5652,U3&p=40884.90760...GwVVaCm.xJ.OtXrTB6n9TuzNfE,UP}
      </attr>
    </attrs>
  </content>
</entry>
```

Related Information

Converting a logon token from XML-encoded text [page 46]
To get a logon token from a serialized session or session token [page 41]

7.4.5 /v1//logon/trustedx509

GET http://<baseURL>/v1/logon/trustedx509

Make a `GET` request to `/v1/logon/trustedx509` to receive a logon token.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/logon/trustedx509`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
  Query Parameter:`?clienttype=<clienttype>`

  > **Note**
  >
  > If any BOE client (Analysis Office, Xcelsius, LiveOffice etc.) using RESTful web services wants to be audited by its `<Clienttype>`, then in the client, while creating the login session, set the `<ClientType>` attribute (as shown above in "Query parameter") . This particular `<Clienttype>` is saved in the auditing database's `<ADS_EVENT>` table under `<Client_type>`column.

- Header:

Name	Value	Note
`Accept`	`application/xml`	Used by default, so use of this header is not necessary.

- Request Body: None

- Response Header

Name	Value	Example
Status Code	HTTP response code.	200 OK
Server	Type of server.	Apache-Coyote/1.1
X-SAP-LogonToken	Returned encoded token.	COMMANDCOM-LCM: 6400@{3&2=588...9QO0XnE,UP}
Date	Date and time of response.	Fri, 16 Dec 2011 22:00:57 GMT
Content-Type	Type of content in the response body.	application/xml

Response Body:

```
<?xml version="1.0" ?>
<entry xmlns="http://www.w3.org/2005/Atom">
  <author>
    <name>
      @COMMANDCOM-LCM:6400
    </name>
  </author>
  <id>
    tag:sap.com,2010:bip-rs/logon/trusted
  </id>
  <title type="text">
    Logon Result
  </title>
  <updated>
    2011-12-07T21:46:57.091Z
  </updated>
  <content type="application/xml">
    <attrs xmlns="http://www.sap.com/rws/bip">
      <attr name="logonToken" type="string">
        COMMANDCOM-LCM:
6400@{3&2=5652,U3&p=40884.90760...GwVVaCm.xJ.OtXrTB6n9TuzNfE,UP}
      </attr>
    </attrs>
  </content>
</entry>
```

> **Note**
>
> For the API to work as expected, your **WACS server must be configured with HTTPS/SSL**. For more information, refer to the topic *To Configure HTTPS/SSL* available at https://uacp2.hana.ondemand.com/viewer/2e167338c1b24da9b2a94e68efd79c42/4.2.4/en-US/469a2b9f6e041014910aba7db0e91070.html

7.4.6 /v1/logoff

POST http://<baseURL>/v1/logoff

Make a `POST` request to `/v1/logoff` to invalidate the logon token and log off the BI platform.

Request:

- Method: POST
- URL: `http://<baseURL>/v1/logoff`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: none

Related Information

To log off the BI platform [page 49]

7.5 Managing Scheduling

This section provides you information on Scheduling RESTful APIs. Using these APIs, you can perform the following:

- Getting template for scheduling
- Creating schedule now
- Creating schedule once
- Creating hourly schedule
- Creating daily schedule
- Creating weekly schedule
- Creating monthly schedule
- Creating schedule for nth day of month
- Creating schedule for first monday of the month

- Creating schedule for calendar
- Creating schedule for xth day of nth week of month
- Creating schedule for last day of the month
- Getting instances for a report
- Getting instances details of a report
- Getting schedule list for a report
- Getting details of a schedule
- Getting instances details of a schedule
- Sorting and Filtering

> **Note**
>
> - The Start Date and End Date format in the Request Page for Scheduling APIs is in GMT+00 format.
> - If you enter invalid, null, or empty values for the Start Date and End Date fields, system displays a generic error, which is *"Input params are not valid"*. However, we recommend you to validate and enter a valid Start Date and End Date fields.

7.5.1 Getting Template for Scheduling

GET http://<baseURL>/v1/<path>/<id>/schedules/schedule

Get a list of URLs that can be used to schedule a resource by sending a request to `v1/<path>/<id>/schedules/schedule` using the `GET` method.

> **Note**
>
> For `<path>`, you can define the following values:
>
> - infostore
> - documents
> - publications

> **Note**
>
> If the resource is not schedulable, an error is returned.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule`
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
Accept	application/xml
X-SAP-LogonToken	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  An XML feed of scheduling URLs. This example shows the scheduling URLs for the resource with ID=6148.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <entry>
        <title type="text">now</title>
        <link href="http://localhost:6405/biprws/v1/infostore/6148/schedules/
schedule/now"/>
    </entry>
    <entry>
        <title type="text">once</title>
        <link href="http://localhost:6405/biprws/v1/infostore/6148/schedules/
schedule/once"/>
    </entry>
    <entry>
        <title type="text">hourly</title>
        <link href="http://localhost:6405/biprws/v1/infostore/6148/schedules/
schedule/hourly"/>
    </entry>
    <entry>
        <title type="text">daily</title>
        <link href="http://localhost:6405/biprws/v1/infostore/6148/schedules/
schedule/daily"/>
    </entry>
    <entry>
        <title type="text">weekly</title>
        <link href="http://localhost:6405/biprws/v1/infostore/6148/schedules/
schedule/weekly"/>
    </entry>
    <entry>
        <title type="text">monthly</title>
        <link href="http://localhost:6405/biprws/v1/infostore/6148/schedules/
schedule/monthly"/>
    </entry>
    <entry>
        <title type="text">firstmondayofmonth</title>
```

```
        <link href="http://localhost:6405/biprws/v1/infostore/6148/schedules/
schedule/firstmondayofmonth"/>
    </entry>
    <entry>
        <title type="text">lastdayofmonth</title>
        <link href="http://localhost:6405/biprws/v1/infostore/6148/schedules/
schedule/lastdayofmonth"/>
    </entry>
    <entry>
        <title type="text">calendar</title>
        <link href="http://localhost:6405/biprws/v1/infostore/6148/schedules/
schedule/calendar"/>
    </entry>
    <entry>
        <title type="text">xthdaynthweekofmonth</title>
        <link href="http://localhost:6405/biprws/v1/infostore/6148/schedules/
schedule/xthdaynthweekofmonth"/>
    </entry>
    <entry>
        <title type="text">nthdayofmonth</title>
        <link href="http://localhost:6405/biprws/v1/infostore/6148/schedules/
schedule/nthdayofmonth"/>
    </entry>
</feed>
```

7.5.2 Creating Schedule now

Schedule a resource by making requests to the `/v1/<path>/<id>/schedules/schedule/now` URL.

> **Note**
>
> For `<path>`, you can define the following values:
>
> - infostore
> - documents
> - publications

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to schedule a resource.

GET http://<baseURL>/v1/<path>/<id>/schedules/schedule/now

Make a `GET` request to `/v1/<path>/<id>/schedules/schedule/now` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/now`
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.

- Header:

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="retriesallowed" type="int32">0</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
        </attrs>
    </content>
</entry>
```

POST http://<baseURL>/v1/<path>/<id>/schedules/schedule/now

Make a `POST` request to `/v1/<path>/<id>/schedules/schedule/now` to schedule a resource

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/now`.
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource. Replace `<form>` with the scheduling method, for example, now, daily, weekly, or now.
- Header:

Name	Value
Content-Type	application/xml
Accept	application/xml
X-SAP-LogonToken	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="retriesallowed" type="int32">2</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
        </attrs>
    </content>
</entry>
```

  - Use`<attr name="retriesallowed" type="int32">2</attr>` to define number of retries allowed.
  - Use`<attr name="retryintervalinseconds" type="int32">1800</attr>` to define the retry interval in milliseconds.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
Location	A URL that contains a link to the location of the scheduled instance.
Date	The date and time of the response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: none

7.5.3 Creating Schedule Once

Schedule a resource by making requests to the `/v1/<path>/<id>/schedules/schedule/once` URL.

> **Note**
>
> For `<path>`, you can define the following values:
>
> - infostore

- documents
- publications

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to schedule a resource.

GET http://<baseURL>/v1/<path>/<id>/schedules/schedule/once

Make a `GET` request to `/v1/<path>/<id>/schedules/schedule/once` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/once`
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</
attr>
            <attr name="retriesallowed" type="int32">0</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
```

```
        </attrs>
    </content>
</entry>
```

POST http://<baseURL>/v1/<path>/<id>/schedules/schedule/once

Make a `POST` request to `v1/<path>/<id>/schedules/schedule/once` to schedule a resource

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/once`.
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</
attr>
            <attr name="retriesallowed" type="int32">2</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
        </attrs>
    </content>
</entry>
```

  - Use`<attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</attr>` to define start time.
  - Use`<attr name="retriesallowed" type="int32">2</attr>` to define number of retries allowed.
  - Use`<attr name="retryintervalinseconds" type="int32">1800</attr>` to define the retry interval in milliseconds.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
Location	A URL that contains a link to the location of the scheduled instance.
Date	The date and time of the response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: none

7.5.4 Creating Hourly Schedule

Schedule a resource by making requests to the `/v1/<path>/<id>/schedules/schedule/hourly` URL.

> **Note**
>
> For `<path>`, you can define the following values:
>
> - infostore
> - documents
> - publications

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to schedule a resource.

GET http://<baseURL>/v1/<path>/<id>/schedules/schedule/hourly

Make a `GET` request to `/v1/<path>/<id>/schedules/schedule/hourly` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/hourly`
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="hours" type="int32">0</attr>
            <attr name="minutes" type="int32">0</attr>
            <attr name="retriesallowed" type="int32">0</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
            <attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</
attr>
        </attrs>
    </content>
</entry>
```

POST http://<baseURL>/v1/<path>/<id>/schedules/schedule/hourly

Make a `POST` request to `v1/<path>/<id>/schedules/schedule/hourly` to schedule a resource

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/hourly`.
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="hours" type="int32">1</attr>
            <attr name="minutes" type="int32">30</attr>
            <attr name="retriesallowed" type="int32">2</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
            <attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</
attr>
        </attrs>
    </content>
</entry>
```

  Fill in each `<attr>` element with an appropriate value for your scheduling request. For example,
  - Use `<attr name="hours" type="int32">1</attr>` to define the hours
  - Use `<attr name="minutes" type="int32">30</attr>` to define the minutes
  - Use`<attr name="retriesallowed" type="int32">2</attr>` to define number of retries allowed.
  - Use`<attr name="retryintervalinseconds" type="int32">1800</attr>` to define the retry interval in milliseconds.

.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
Location	A URL that contains a link to the location of the scheduled instance.
Date	The date and time of the response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: none

7.5.5 Creating Daily Schedule

Schedule a resource by making requests to the `/v1/<path>/<id>/schedules/schedule/daily` URL.

> **Note**
>
> For `<path>`, you can define the following values:

- infostore
- documents
- publications

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to schedule a resource.

GET http://<baseURL>/v1/<path>/<id>/schedules/schedule/daily

Make a `GET` request to `/v1/<path>/<id>/schedules/schedule/daily` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/daily`
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Geo Analysis Demo</title>
    <updated>2016-06-15T06:02:47.603Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="everyndays" type="int32">0</attr>
            <attr name="retriesallowed" type="int32">0</attr>
```

```
            <attr name="retryintervalinseconds" type="int32">1800</attr>
            <attr name="endtime" type="datetime">2022-12-19T16:39:58.250Z</attr>
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.250Z</
attr>
        </attrs>
    </content>
</entry>
```

POST http://<baseURL>/v1/<path>/<id>/schedules/schedule/daily

Make a `POST` request to `v1/<path>/<id>/schedules/schedule/daily` to schedule a resource

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/daily`.
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="everyndays" type="int32">25</attr>
            <attr name="retriesallowed" type="int32">2</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
            <attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</
attr>
        </attrs>
    </content>
</entry>
```

Fill in each `<attr>` element with an appropriate value for your scheduling request. For example,

- Use`<attr name="everyndays" type="int32">25</attr>` to define the number of days.
- Use`<attr name="retriesallowed" type="int32">2</attr>` to define number of retries allowed.
- Use`<attr name="retryintervalinseconds" type="int32">1800</attr>` to define the retry interval in milliseconds.
- Use`<attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>` to define end time.

- Use<attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</attr> to define start time.

.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
Location	A URL that contains a link to the location of the scheduled instance.
Date	The date and time of the response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: none

7.5.6 Creating Weekly Schedule

Schedule a resource by making requests to the `/v1/<path>/<id>/schedules/schedule/weekly` URL.

> **Note**
>
> For `<path>`, you can define the following values:
>
> - infostore
> - documents
> - publications

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to schedule a resource.

GET http://<baseURL>/v1/<path>/<id>/schedules/schedule/weekly

Make a `GET` request to `/v1/<path>/<id>/schedules/schedule/weekly` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/weekly`
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.

- Header:

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="sunday" type="bool">false</attr>
            <attr name="saturday" type="bool">false</attr>
            <attr name="tuesday" type="bool">false</attr>
            <attr name="wednesday" type="bool">false</attr>
            <attr name="thursday" type="bool">false</attr>
            <attr name="friday" type="bool">false</attr>
            <attr name="monday" type="bool">false</attr>
            <attr name="retriesallowed" type="int32">0</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
            <attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</
attr>
        </attrs>
    </content>
</entry>
```

POST http://<baseURL>/v1/<path>/<id>/schedules/schedule/weekly

Make a `POST` request to `v1/<path>/<id>/schedules/schedule/weekly` to schedule a resource

Request:

- Method: `POST`

- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/weekly`.
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="sunday" type="bool">false</attr>
            <attr name="saturday" type="bool">false</attr>
            <attr name="tuesday" type="bool">true</attr>
            <attr name="wednesday" type="bool">true</attr>
            <attr name="thursday" type="bool">false</attr>
            <attr name="friday" type="bool">false</attr>
            <attr name="monday" type="bool">false</attr>
            <attr name="retriesallowed" type="int32">2</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
            <attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</
attr>
        </attrs>
    </content>
</entry>
```

  Fill in each `<attr>` element with an appropriate value for your scheduling request. For example,
  - Use`<attr name="<weekday>" type="bool">false</attr>` to define the week days ranging from Sunday through Saturday, set the value to true or false.
  - Use`<attr name="retriesallowed" type="int32">2</attr>` to define number of retries allowed.
  - Use`<attr name="retryintervalinseconds" type="int32">1800</attr>` to define the retry interval in milliseconds.
  - Use`<attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>` to define end time.
  - Use`<attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</attr>` to define start time.

.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
Location	A URL that contains a link to the location of the scheduled instance.
Date	The date and time of the response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: none

7.5.7 Creating Monthly Schedule

Schedule a resource by making requests to the `/v1/<path>/<id>/schedules/schedule/monthly` URL.

> **Note**
>
> For `<path>`, you can define the following values:
>
> - infostore
> - documents
> - publications

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to schedule a resource.

GET http://<baseURL>/v1/<path>/<id>/schedules/schedule/monthly

Make a `GET` request to `/v1/<path>/<id>/schedules/schedule/monthly` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/monthly`
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="everynmonths" type="int32">0</attr>
            <attr name="retriesallowed" type="int32">0</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
            <attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</
attr>
        </attrs>
    </content>
</entry>
```

POST http://<baseURL>/v1/<path>/<id>/schedules/schedule/monthly

Make a `POST` request to `v1/<path>/<id>/schedules/schedule/monthly` to schedule a resource

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/monthly`.
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="everynmonths" type="int32">5</attr>
            <attr name="retriesallowed" type="int32">2</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
            <attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</
attr>
        </attrs>
    </content>
</entry>
```

  Fill in each `<attr>` element with an appropriate value for your scheduling request. For example,
  - Use`<attr name="everynmonths" type="int32">5</attr>` to define the value of month.
  - Use`<attr name="retriesallowed" type="int32">2</attr>` to define number of retries allowed.
  - Use`<attr name="retryintervalinseconds" type="int32">1800</attr>` to define the retry interval in milliseconds.
  - Use`<attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>` to define end time.
  - Use`<attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</attr>` to define start time.

.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
Location	A URL that contains a link to the location of the scheduled instance.
Date	The date and time of the response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: none

7.5.8 Creating Schedule for Nth Day of Month

Schedule a resource by making requests to the `/v1/<path>/<id>/schedules/schedule/nthdayofmonth` URL.

> **Note**
>
> For `<path>`, you can define the following values:
>
> - infostore
> - documents
> - publications

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to schedule a resource.

GET http://<baseURL>/v1/<path>/<id>/schedules/schedule/nthdayofmonth

Make a `GET` request to `/v1/<path>/<id>/schedules/schedule/nthdayofmonth` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/nthdayofmonth`
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
```

```
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="nthday" type="int32">0</attr>
            <attr name="retriesallowed" type="int32">0</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
            <attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</
attr>
        </attrs>
    </content>
</entry>
```

POST http://<baseURL>/v1/<path>/<id>/schedules/schedule/nthdayofmonth

Make a `POST` request to `v1/<path>/<id>/schedules/schedule/nthdayofmonth` to schedule a resource

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/nthdayofmonth`.
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="nthday" type="int32">5</attr>
            <attr name="retriesallowed" type="int32">2</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
            <attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</
attr>
        </attrs>
    </content>
</entry>
```

Fill in each `<attr>` element with an appropriate value for your scheduling request. For example,

- Use`<attr name="nthday" type="int32">5</attr>` to define the value for the nth day of the month.

- Use`<attr name="retriesallowed" type="int32">2</attr>` to define number of retries allowed.
- Use`<attr name="retryintervalinseconds" type="int32">1800</attr>` to define the retry interval in milliseconds.
- Use`<attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>` to define end time.
- Use`<attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</antr>` to define start time.

.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
Location	A URL that contains a link to the location of the scheduled instance.
Date	The date and time of the response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: none

7.5.9 Creating Schedule for First Monday of the Month

Schedule a resource by making requests to the `/v1/<path>/<id>/schedules/schedule/firstmondayofmonth` URL.

> **Note**
>
> For `<path>`, you can define the following values:
>
> - infostore
> - documents
> - publications

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to schedule a resource.

GET http://<baseURL>/v1/<path>/<id>/schedules/schedule/firstmondayofmonth

Make a `GET` request to `/v1/<path>/<id>/schedules/schedule/firstmondayofmonth` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/firstmondayofmonth`
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://10.160.206.105:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="retriesallowed" type="int32">0</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
            <attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</
attr>
        </attrs>
    </content>
</entry>
```

POST http://<baseURL>/v1/<path>/<id>/schedules/schedule/firstmondayofmonth

Make a `POST` request to `v1/<path>/<id>/schedules/schedule/firstmondayofmonth` to schedule a resource

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/firstmondayofmonth`. Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://10.160.206.105:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="retriesallowed" type="int32">2</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
            <attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</
attr>
        </attrs>
    </content>
</entry>
```

  Fill in each `<attr>` element with an appropriate value for your scheduling request. For example,
  - Use`<attr name="retriesallowed" type="int32">2</attr>` to define number of retries allowed.
  - Use`<attr name="retryintervalinseconds" type="int32">1800</attr>` to define the retry interval in milliseconds.
  - Use`<attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>` to define end time.
  - Use`<attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</attr>` to define start time.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
Location	A URL that contains a link to the location of the scheduled instance.
Date	The date and time of the response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: none

7.5.10 Creating Schedule for Calendar

Schedule a resource by making requests to the `/v1/<path>/<id>/schedules/schedule/calendar` URL.

> **Note**
>
> For `<path>`, you can define the following values:
>
> - infostore
> - documents
> - publications

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to schedule a resource.

GET http://<baseURL>/v1/<path>/<id>/schedules/schedule/calendar

Make a `GET` request to `/v1/<path>/<id>/schedules/schedule/calendar` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/calendar`
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://10.160.206.105:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="calendarid" type="int32">0</attr>
            <attr name="retriesallowed" type="int32">0</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
        </attrs>
    </content>
</entry>
```

POST http://<baseURL>/v1/<path>/<id>/schedules/schedule/calendar

Make a `POST` request to `v1/<path>/<id>/schedules/schedule/calendar` to schedule a resource

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/calendar`.
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
```

```
    <author>
        <name>Administrator</name>
        <uri>http://10.160.206.105:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="calendarid" type="int32">6705</attr>
            <attr name="retriesallowed" type="int32">2</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
        </attrs>
    </content>
</entry>
```

Fill in each `<attr>` element with an appropriate value for your scheduling request. For example,

- Use`<attr name="calendarid" type="int32">6705</attr>` to define the calendar id.
- Use`<attr name="retriesallowed" type="int32">2</attr>` to define number of retries allowed.
- Use`<attr name="retryintervalinseconds" type="int32">1800</attr>` to define the retry interval in milliseconds.

.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
Location	A URL that contains a link to the location of the scheduled instance.
Date	The date and time of the response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: none

7.5.11 Creating Schedule for Xth Day of Nth Week of a Month

Schedule a resource by making requests to the `/v1/<path>/<id>/schedules/schedule/xthdaynthweekofmonth` URL.

> **Note**
>
> For `<path>`, you can define the following values:
>
> - infostore
> - documents
> - publications

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to schedule a resource.

GET http://<baseURL>/v1/<path>/<id>/schedules/schedule/xthdaynthweekofmonth

Make a `GET` request to `/v1/<path>/<id>/schedules/schedule/xthdaynthweekofmonth` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/xthdaynthweekofmonth`
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="nthday" type="int32">0</attr>
            <attr name="nthweek" type="int32">0</attr>
            <attr name="retriesallowed" type="int32">0</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
            <attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</
attr>
```

```
        </attrs>
    </content>
</entry>
```

POST http://<baseURL>/v1/<path>/id/schedules/schedule/xthdaynthweekofmonth

Make a `POST` request to `v1/<path>/<id>/schedules/schedule/xthdaynthweekofmonth` to schedule a resource

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/xthdaynthweekofmonth`. Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="nthday" type="int32">5</attr>
            <attr name="nthweek" type="int32">4</attr>
            <attr name="retriesallowed" type="int32">2</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
            <attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</
attr>
        </attrs>
    </content>
</entry>
```

Fill in each `<attr>` element with an appropriate value for your scheduling request. For example,

- Use`<attr name="nthday" type="int32">5</attr>` to define nth day ranging from 0 to 7.
- Use`<attr name="nthweek" type="int32">4</attr>` to define nth week ranging from 0 to 5.
- Use`<attr name="retriesallowed" type="int32">2</attr>` to define number of retries allowed.
- Use`<attr name="retryintervalinseconds" type="int32">1800</attr>` to define number of retry intervals in milliseconds.

- Use`<attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>` to define end time.
- Use`<attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</attr>` to define start time.

.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
Location	A URL that contains a link to the location of the scheduled instance.
Date	The date and time of the response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: none

7.5.12 Creating Schedule for Last Day of the Month

Schedule a resource by making requests to the `/v1/<path>/<id>/schedules/schedule/lastdayofmonth` URL.

> **Note**
>
> For `<path>`, you can define the following values:
>
> - infostore
> - documents
> - publications

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to schedule a resource.

GET http://<baseURL>/v1/<path>/<id>/schedules/schedule/lastdayofmonth

Make a `GET` request to `/v1/<path>/<id>/schedules/schedule/lastdayofmonth` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET

- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/lastdayofmonth`
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.
- Header:

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="retriesallowed" type="int32">0</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
            <attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</
attr>
        </attrs>
    </content>
</entry>
```

POST http://<baseURL>/v1/<path>/<id>/schedules/schedule/lastdayofmonth

Make a `POST` request to `v1/<path>/<id>/schedules/schedule/lastdayofmonth` to schedule a resource

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/schedule/lastdayofmonth`.

Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID or CUID of a schedulable resource.

- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">Drill Demo</title>
    <updated>2016-06-15T06:02:47.494Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="retriesallowed" type="int32">2</attr>
            <attr name="retryintervalinseconds" type="int32">1800</attr>
            <attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>
            <attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</
attr>
        </attrs>
    </content>
</entry>
```

Fill in each `<attr>` element with an appropriate value for your scheduling request. For example,

- Use`<attr name="retriesallowed" type="int32">2</attr>` to define number of retries allowed.
- Use`<attr name="retryintervalinseconds" type="int32">1800</attr>` to define number of retry intervals in milliseconds.
- Use`<attr name="endtime" type="datetime">2022-12-19T16:39:58.244Z</attr>` to define end time.
- Use`<attr name="starttime" type="datetime">2012-12-19T16:39:58.244Z</attr>` to define start time.

.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
Location	A URL that contains a link to the location of the scheduled instance.
Date	The date and time of the response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: none

7.5.13 Getting Instances for a Report

GET http://<baseURL>/v1/<path>/<id>/instances

You can list instances by sending a request to `/v1/<path>/<id>/instances`, using the `GET` method.

> **Note**
>
> For `<path>`, you can define the following values:
>
> - infostore
> - documents
> - publications

> **Note**
>
> If the resource is not schedulable, an error is returned.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/<path>/<id>/instances`
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID of a scheduled resource.
- Header:

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:

This example shows the details of schedule instances for the resource with ID=6148.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Schedules</title>
    <updated>2016-06-17T07:09:12.875Z</updated>
    <link href="http://localhost:6405/biprws/v1/infostore/6148/instances?
page=1&amp;pagesize=50" rel="self"/>
    <link href="http://localhost:6405/biprws/v1/infostore/6148/instances?
page=1&amp;pagesize=50" rel="first"/>
    <link href="http://localhost:6405/biprws/v1/infostore/6148/instances?
page=1&amp;pagesize=50" rel="last"/>
    <entry>
        <title type="text">Drill Demo</title>
        <link href="http://localhost:6405/biprws/v1/infostore/6148/instances/
6602"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="errormessage" type="string"></attr>
                <attr name="schedulestatus" type="string">9</attr>
                <attr name="uistatus" type="string">Pending</attr>
                <attr name="cuid" type="string">AV60GpT7ajVEpJtlZaie2pI</attr>
                <attr name="name" type="string">Drill Demo</attr>
                <attr name="nextruntime" type="string">Thu Jun 16 21:31:13 PDT
2016</attr>
                <attr name="id" type="string">6602</attr>
                <attr name="type" type="string">Webi</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">Drill Demo</title>
        <link href="http://localhost:6405/biprws/v1/infostore/6148/instances/
6603"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="errormessage" type="string"></attr>
                <attr name="schedulestatus" type="string">9</attr>
                <attr name="uistatus" type="string">Pending</attr>
                <attr name="cuid" type="string">AQyS4kjnjlxBooAkhTzpfBU</attr>
                <attr name="name" type="string">Drill Demo</attr>
                <attr name="nextruntime" type="string">Thu Jun 16 21:31:27 PDT
2016</attr>
                <attr name="id" type="string">6603</antr>
                <attr name="type" type="string">Webi</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.5.14 Getting Instance Details of a Report

GET http://<baseURL>/v1/<path>/<id>/instances/<instance_id>

You can list details of instances by sending a request to `/v1/<path>/<id>/instances/<instance_id>`, using the `GET` method.

> **Note**
>
> For `<path>`, you can define the following values:
>
> - infostore
> - documents
> - publications

> **Note**
>
> If the resource is not schedulable, an error is returned.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/<path>/<id>/instances/<instance_id>`
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID of a scheduled resource.
- Header:

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  This example shows the details of schedule instances for the resource with ID=6602.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Instances</title>
    <updated>2016-06-17T07:13:46.735Z</updated>
    <link href="http://localhost5:6405/biprws/v1/infostore/6148/instances/6602"/>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="owner" type="string">Administrator</attr>
            <attr name="schedulestatus" type="int32">9</attr>
            <attr name="retriesallowed" type="int32">0</attr>
            <attr name="scheduletype" type="string">daily</attr>
            <attr name="cuid" type="string">AV60GpT7ajVEpJtlZaie2pI</attr>
            <attr name="created" type="string">Thu Jun 16 21:31:13 PDT 2016</
attr>
```

```
            <attr name="parentcuid" type="string">AS9ukIRdciZLuUS6ESGVRBg</attr>
            <attr name="endtime" type="string">Mon Dec 19 08:39:58 PST 2022</
attr>
            <attr name="nextruntime" type="string">Thu Jun 16 21:31:13 PDT 2016</
attr>
            <attr name="type" type="string">Webi</attr>
            <attr name="ownerid" type="string">12</attr>
            <attr name="parentid" type="string">6148</attr>
            <attr name="name" type="string">Drill Demo</attr>
            <attr name="id" type="string">6602</attr>
            <attr name="retriesinterval" type="int32">1800</attr>
        </attrs>
    </content>
</entry>
```

Note

With effect from BI 4.2, Support Package **5**, the `schedule status` of the document can have the following two additional values:

- `Expired` (Code=13)
- `Warning` (Code=14)

For reference: Valid Values for 'Schedule Status' with Details

Schedule Status	Code	Meaning
Running	0	Instance is running on the server.
Success	1	Instance successfully ran on the server.
Failed	3	Instance failed on the server.
Paused	8	Instance was paused while running.
Pending	9	Instance has been scheduled but remains in a queue on the server and had not been picked up yet for running.
Recurring	12	When a document has been scheduled at a frequency such as hourly, daily, weekly, monthly, quarterly, weekly, yearly etc., it's instances display the 'recurring' status.
Expired	13	When the 'end-time' of a scheduled document instance is past, but the instance continues to exist due to 'Paused' status on the server, the instance automatically gets 'expired'.
Warning	14	When data in the instance has gotten corrupted due to partial refresh or deletion on the server, the schedule status displays a 'warning'.

7.5.15 Getting Schedule List for a Report

GET http://<baseURL>/v1/<path>/<id>/schedules

You can list schedules for a particular publication by sending a request to `/v1/<path>/<id>/schedules`, using the `GET` method.

> **Note**
>
> For `<path>`, you can define the following values:
>
> - infostore
> - documents
> - publications

> **Note**
>
> If the resource is not schedulable, an error is returned.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/<path>/<id>/schedules`
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID of a scheduled resource.
- Header:

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:

This example shows the details of schedule instances for the resource with ID=6602.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Schedules</title>
    <updated>2016-06-17T07:46:16.867Z</updated>
    <link href="http://localhost:6405/biprws/v1/infostore/6148/schedules?
page=1&amp;pagesize=50" rel="self"/>
    <link href="http://localhost:6405/biprws/v1/infostore/6148/schedules?
page=1&amp;pagesize=50" rel="first"/>
    <link href="http://localhost:6405/biprws/v1/infostore/6148/schedules?
page=1&amp;pagesize=50" rel="last"/>
    <entry>
        <title type="text">Drill Demo</title>
        <link href="http://localhost:6405/biprws/v1/infostore/6148/schedules/
6602"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="owner" type="string">Administrator</attr>
                <attr name="cuid" type="string">AV60GpT7ajVEpJtlZaie2pI</attr>
                <attr name="created" type="string">Thu Jun 16 21:31:13 PDT 2016</
attr>
                <attr name="parentcuid" type="string">AS9ukIRdciZLuUS6ESGVRBg</
attr>
                <attr name="name" type="string">Drill Demo</attr>
                <attr name="id" type="string">6602</attr>
                <attr name="type" type="string">Webi</attr>
                <attr name="ownerid" type="int32">12</attr>
                <attr name="parentid" type="string">6148</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.5.16 Getting Instances Details of a Schedule

GET http://<baseURL>/v1/<path>/<id>/schedules/<schedule_id>/instances

You can list details of scheduled instances by sending a request to `/v1/<path>/<id>/schedules/<schedule_id>/instances` , using the `GET` method.

> **Note**
>
> For `<path>`, you can define the following values:
>
> - infostore
> - documents
> - publications

> **Note**
>
> If the resource is not schedulable, an error is returned.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/<schedule_id>/instances`
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID of a scheduled resource.
- Header:

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  This example shows the details of schedule instances for the resource with ID=5427 and schedule ID=6263.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Instances</title>
    <updated>2016-06-17T08:32:58.903Z</updated>
    <link href="http://localhost:6405/biprws/v1/documents/5427/schedules/6262/
instances?page=1&amp;pagesize=50" rel="self"/>
    <link href="http://localhost:6405/biprws/v1/documents/5427/schedules/6262/
instances?page=1&amp;pagesize=50" rel="first"/>
    <link href="http://localhost:6405/biprws/v1/documents/5427/schedules/6262/
instances?page=1&amp;pagesize=50" rel="last"/>
    <entry>
        <title type="text">Input Controls And Charts</title>
        <link href="http://localhost:6405/biprws/v1/documents/5427/schedules/
6262/instances/6263"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="owner" type="string">Administrator</attr>
                <attr name="cuid" type="string">AUk.yKLO3i5EvLUw9YrXcH4</attr>
                <attr name="created" type="string">Fri Jun 17 10:29:36 CEST
2016</attr>
                <attr name="parentcuid" type="string">AVOEBpPXOvpGhcrzzE_RAeY</
attr>
                <attr name="name" type="string">Input Controls And Charts</attr>
                <attr name="id" type="string">6263</attr>
                <attr name="type" type="string">Webi</attr>
                <attr name="ownerid" type="int32">12</attr>
                <attr name="parentid" type="string">5427</attr>
            </attrs>
        </content>
    </entry>
```

```
</feed>
```

7.5.17 Sorting and Filtering

Sorting schedules

Sorting is the process of arranging the schedules in a systematic order.

You can now sort schedules sequentially in ascending or descending order based on the following attributes:

- Name
- Description
- Id of the schedule

Filtering schedules

You can filter schedules based on the `updated` attribute.

The query parameters for sort and filter is as follows:

Query Parameter

Parameter Name	Parameter Value
Sort	`sort=<+/-><attribute>` **Note** - For ascending order the value is + - For descending order the value is - - By default the schedules are sorted in ascending based on the name. **Example** sort=+name
Filter	`updated=<Start date year-month-dateThh:mm:ss.Timezone> , <End date year-month-dateThh:mm:ss.Timezone>` **Note** If end date is not specified, by default current date will be considered. **Example** updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z

Parameter Name	Parameter Value
Paging	• `page=<page number>` • `pagesize=<number>` `pagesize` is the number of objects displayed in response **Note** By default `page=1` and `pagesize=50` that is, in one page number 1, the top 50 objects will be displayed in response.

URL: `GET http://<baseURL>/v1/<path>/id/schedules/schedule?sort=<name>&updated=<Start year-month-dateThh:mm:ss.727Z> , <End year-month-dateThh:mm:ss.727Z>&page=<page number>&pagesize=<number>`

Example

URL: `GET http://<baseURL>/v1/infostore/id/schedules/schedule?sort=+name&updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z&page=2&pagesize=5`

once you send the request, the response displays the second page which has five objects and the list of schedules updated between the specified date and sorted in ascending order.

Related Information

To use pagination with results [page 63]

7.5.18 Getting Details of a Schedule

GET http://<baseURL>/v1/<path>/<id>/schedules/<schedule_id>

You can list the details of a schedule by sending a request to `/v1/<path>/<id>/schedules/<schedule_id>`, using the `GET` method.

Note

For `<path>`, you can define the following values:

- infostore
- documents
- publications

> **Note**
>
> If the resource cannot be scheduled, an error is returned.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/<path>/<id>/schedules/<schedule_id>`
  Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<id>` with the ID of a scheduled resource.
- Header:

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  This example shows the details of schedule instances for the resource with ID=6602.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Schedules</title>
    <updated>2016-06-17T07:52:24.836Z</updated>
    <link href="http://localhost:6405/biprws/v1/infostore/6148/schedules/6602"/>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="owner" type="string">Administrator</attr>
            <attr name="schedulestatus" type="int32">9</attr>
            <attr name="retriesallowed" type="int32">0</attr>
            <attr name="scheduletype" type="string">weekly</attr>
            <attr name="cuid" type="string">AV60GpT7ajVEpJtlZaie2pI</attr>
            <attr name="created" type="string">Thu Jun 16 21:31:13 PDT 2016</
attr>
            <attr name="parentcuid" type="string">AS9ukIRdciZLuUS6ESGVRBg</attr>
            <attr name="endtime" type="string">Mon Dec 19 08:39:58 PST 2022</
attr>
            <attr name="nextruntime" type="string">Thu Jun 16 21:31:13 PDT 2016</
attr>
            <attr name="type" type="string">Webi</attr>
            <attr name="ownerid" type="string">12</attr>
            <attr name="parentid" type="string">6148</attr>
            <attr name="name" type="string">Drill Demo</attr>
            <attr name="id" type="string">6602</attr>
            <attr name="retriesinterval" type="int32">1800</attr>
```

```
        </attrs>
    </content>
</entry>
```

> **Note**
>
> With effect from BI 4.2, Support Package **4**, the schedule type of the document (such as 'daily', 'hourly', 'weekly' etc.) is displayed explicitly in the response.

7.5.19 Finding Available Server Groups for Scheduling

You can fetch the available server group details for scheduling documents.

GET `http://<baseURL>/v1/servergroups`

- URL: `http://<baseURL>/v1/servergroups`

  > **Note**
  >
  > Replace `<baseURL>` with the base URL for RESTful web service requests (such as `host:<port>/biprws`).

- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none
  Below is an example of the response received from the server:

```
{
   "entries": [
    {
      "cuid": "FlLSkVgbaQkAYhIAAACnB.sAAFBWATGQ",
      "created": "Feb 1, 2017 5:49 PM",
      "name": "Bob's group",
      "description": "My group of reporting servers",
      "exclusive": "true",
      "id": "11133",
      "updated": "Feb 1, 2017 5:50 PM"
    }
  ]
}
```

7.5.20 Fetching Available Calendars and Calendar Dates

Using RESTful Web service APIs, you can fetch available calendars and the dates within those calendars.

Fetching Available Calendars

GET `http://<baseURL>/v1/calendars`

Use the `GET` method with the below URL to fetch the available calendars:

- URL: `http://<baseURL>/v1/calendars`

  > **Note**
  >
  > Replace `<baseURL>` with the base URL for RESTful web service requests (such as `host:<port>/biprws`).

- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none
  Below is an example of the response received from the server:

```
{
  "entries": [
    {
      "cuid": "Fo3LkVj_sgUAYhIAAAB3gAAAAFBWATGQ",
      "created": "Feb 1, 2017 5:20 PM",
      "name": "Bob's calendar",
      "description": "My calendar",
      "id": "10588",
      "updated": "Feb 1, 2017 5:21 PM"
    }
  ]
}
```

Fetching Available Calendar Dates

GET `http://<baseURL>/v1/calendars/<calendar_ID>/dates`

Use the `GET` method with the below URL to fetch the available calendars:

- URL: `http://<baseURL>/v1/calendars/<calendar_ID>/dates`

> **Note**
>
> Replace `<baseURL>` with the base URL for RESTful web service requests (such as `host:<port>/biprws`).

Where, `<calendar_ID>` is the ID of your calendar. An example of the URL is: `http://10.200.197.400:6405/biprws/v1/calendars/10588/dates`

- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none
  Below is an example of the response received from the server:

```
{
 "entries": [
    {
      "date": "01-02-2017"
    },
    {
      "date": "10-02-2017"
    },
    {
      "date": "24-02-2017"
    },
    {
      "date": "28-02-2017"
    },
    {
      "date": "01-03-2017"
    },
    {
      "date": "31-03-2017"
    }
  ]
}
```

> **Note**
>
> The API **does not return dates for recurring calendars**.

7.6 Managing Users

This section provides you information on RESTful APIs that can be used to manage single user. Using these API's, you can perform the following:

- Listing users
- Creating an user
- Getting user details
- Modifying user details
- Deleting user

> **Note**
>
> Create, Modify and Delete operations can be performed on enterprise user only.

7.6.1 Creating New User

Create a user in BI platform through making request to the `/v1/users/user` URL.

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to create a user on BI platform.

GET http://<baseURL>/v1/users/user

Make a `GET` request to `/v1/users/user` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/users/user`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.

Name	Value
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="password" type="string"></attr>
            <attr name="forcepasswordchange" type="bool">true</attr>
            <attr name="nameduser" type="bool">false</attr>
            <attr name="name" type="string"></attr>
            <attr name="description" type="string"></attr>
            <attr name="fullname" type="string"></attr>
            <attr name="email" type="string"></attr>
            <attr name="passwordexpire" type="bool">false</attr>
        </attrs>
    </content>
</entry>
```

POST http://<baseURL>/v1/users/user

To create a user, use `POST` request to `/v1/users/user` URL.

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/users/user`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="password" type="string"></attr>
            <attr name="forcepasswordchange" type="bool">true</attr>
            <attr name="nameduser" type="bool">false</attr>
            <attr name="name" type="string"></attr>
            <attr name="description" type="string"></attr>
            <attr name="fullname" type="string"></attr>
            <attr name="email" type="string"></attr>
            <attr name="passwordexpire" type="bool">false</attr>
        </attrs>
    </content>
```

```
</entry>
```

- Use `<attr name="password" type="string"></attr>` to define user password.
- Use `<attr name="nameduser" type="bool"></attr>` to define the user license type.
- Use `<attr name="name" type="string"></attr>` to define user name.
- Use `<attr name="description" type="string"></attr>` to define user description.
- Use `<attr name="fullname" type="string"></attr>` to define full name of the user.
- Use `<attr name="email" type="string"></attr>` to define e-mail address.
- Use `<attr name="passwordexpire" type="bool"></attr>` to set the password expiry.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
X-SAP-LogonToken	A logon token.
Date	Date and time of the response
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  The response body contains the details of user created .

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text"></title>
    <updated>2016-05-27T05:54:58.173Z</updated>
    <link href="http://localhost:6405/biprws/v1/users/6704" rel="alternate" />
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">AZKI7NLt.z5MqhjQfhYDpTw</attr>
            <attr name="forcepasswordchange" type="bool">true</attr>
            <attr name="nameduser" type="bool">false</attr>
            <attr name="name" type="string">administrator222</attr>
            <attr name="description" type="string"></attr>
            <attr name="id" type="string">6704</attr>
            <attr name="fullname" type="string"></attr>
            <attr name="email" type="string"></attr>
            <attr name="passwordexpire" type="bool">false</attr>
        </attrs>
    </content>
</entry>
```

7.6.2 Getting User Details

GET http://<baseURL>/v1/users/<user_id>

Get user details using `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/users/<user_id>`

> **Note**
>
> Replace `<baseURL>` with the base URL for RESTful web service requests (such as `host:<port>/biprws`).

- Header:

Name	Value
Content-Type	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  An XML template of user details in BOE system is displayed. This example shows user details for the resource with user Id=12.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">user</title>
    <updated>2016-05-27T05:58:00.440Z</updated>
    <link href="http://localhost:6405/biprws/v1/users/12" rel="alternate" />
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">AfRWaT5_131NlLLf5bRMLKY</attr>
            <attr name="forcepasswordchange" type="bool">false</attr>
            <attr name="parentcuid" type="string">AXhmigik4CBKra9ZYzR2ezE</attr>
```

```
            <attr name="description" type="string">Administrator account</attr>
            <attr name="type" type="string">User</attr>
            <attr name="ownerid" type="string">12</attr>
            <attr name="parentid" type="string">19</attr>
            <attr name="lastlogontime" type="string">Mon Jan 23 03:11:28 PST
2017</attr>
            <attr name="nameduser" type="bool">false</attr>
            <attr name="name" type="string">Administrator</attr>
            <attr name="disabled" type="bool">false</attr>
            <attr name="id" type="string">12</attr>
            <attr name="fullname" type="string"></attr>
            <attr name="inbox" type="int32">0</attr>
            <attr name="updated" type="string">Fri May 27 07:51:46 CEST 2016</
attr>
            <attr name="email" type="string"></attr>
            <attr name="passwordexpire" type="bool">true</attr>
        </attrs>
    </content>
</entry>
```

> **Note**
>
> With effect from **SP4** release of the BI platform SDK, **the last logon time of the user** is also returned in the response body.

7.6.3 Modifying User Details

PUT http://<baseURL>v1/users/<user_id>

To modify a user, use `PUT` request to `/v1/users/<user_id>` URL.

Request:

- Method: `PUT`
- URL: `http://<baseURL>/v1/users/<user_id`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">user</title>
    <updated>2016-05-27T06:03:25.583Z</updated>
    <link href="http://localhost:6405/biprws/v1/users/12" rel="alternate" />
    <content type="application/xml">
```

```
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">AfRWaT5_131NlLLf5bRMLKY</attr>
            <attr name="forcepasswordchange" type="bool">false</attr>
            <attr name="parentcuid" type="string">AXhmigik4CBKra9ZYzR2ezE</attr>
            <attr name="description" type="string">Administrator account</attr>
            <attr name="type" type="string">User</attr>
            <attr name="ownerid" type="string">12</attr>
            <attr name="parentid" type="string">19</attr>
            <attr name="nameduser" type="bool">false</attr>
            <attr name="name" type="string">Administrator211</attr>
            <attr name="disabled" type="bool">false</attr>
            <attr name="id" type="string">12</attr>
            <attr name="fullname" type="string"></attr>
            <attr name="inbox" type="int32">0</attr>
            <attr name="updated" type="string">Fri May 27 08:01:03 CEST 2016</
attr>
            <attr name="email" type="string"></attr>
            <attr name="passwordexpire" type="bool">true</attr>
        </attrs>
    </content>
</entry>
```

- Use `<attr name="forcepasswordchange" type="bool"></attr>` to define user password change.
- Use `<attr name="description" type="string"></attr>` to define a description.
- Use `<attr name="ownerid" type="string"></attr>` to define owner Id.
- Use `<attr name="parentid" type="string"></attr>` to define parent Id.
- Use `<attr name="newPassword" type="string"></attr>` to define password.
- Use `<attr name="nameduser" type="bool"></attr>` to define license type of user.
- Use `<attr name="name" type="string"></attr>` to define the name of the user.
- Use `<attr name="disabled" type="bool"></attr>` to check whether the user account is disabled or not.
- Use `<attr name="fullname" type="string"></attr>` to define the full name of the user.
- Use `<attr name="emailAddress" type="string"></attr>` to define e-mail address.
- Use `<attr name="passwordexpiry" type="bool"></attr>` to set the password expiry.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
X-SAP-LogonToken	A logon token.
Date	Date and time of the response
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  The response body contains the details of modified user.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">user</title>
    <updated>2016-05-27T06:04:29.799Z</updated>
    <link href="http://localhost:6405/biprws/v1/users/12" rel="alternate" />
    <content type="application/xml">
```

```
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">AfRWaT5_131NlLLf5bRMLKY</attr>
            <attr name="forcepasswordchange" type="bool">false</attr>
            <attr name="parentcuid" type="string">AXhmigik4CBKra9ZYzR2ezE</attr>
            <attr name="description" type="string">Administrator account</attr>
            <attr name="type" type="string">User</attr>
            <attr name="ownerid" type="string">12</attr>
            <attr name="parentid" type="string">19</attr>
            <attr name="nameduser" type="bool">false</attr>
            <attr name="name" type="string">Administrator211</attr>
            <attr name="disabled" type="bool">false</attr>
            <attr name="id" type="string">12</attr>
            <attr name="fullname" type="string"></attr>
            <attr name="inbox" type="int32">0</attr>
            <attr name="updated" type="string">Fri May 27 08:01:03 CEST 2016</
attr>
            <attr name="email" type="string"></attr>
            <attr name="passwordexpire" type="bool">true</attr>
        </attrs>
    </content>
</entry>
```

7.6.4 Specifying Locale and Time-zone information for current user

GET `http://<baseURL>/v1/users/locale`

Request:

- Method: **GET**
- URL: `http://<baseURL>/v1/users/locale`
  Replace `<baseURL>` with the base URL for RESTful web service requests (such as `10.10.10.10:<port>/biprws`).
- Header:

Name	Value
`Accept`	`application/xml`
`Content-Type`	`application/xml`
`X-SAP-LogonToken`	The logon token value.

- Body: none

Response:

- Header:

Name	Value
`Accept`	`application/xml`

Name	Value
Content-Type	application/xml
X-SAP-LogonToken	The logon token value.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
  <content type="application/xml">
    <attrs xmlns="http://www.sap.com/rws/bip">
    <attr name="productlocale" type="string"></attr>
    <attr name="timezone" type="string"></attr>
    <attr name="preferredviewinglocale" type="string"></attr>
    </attrs>
 </content>
</entry>
```

> **Note**
>
> **Copy** the body contents.

POST http://<baseURL>/v1/users/locale

Request:

- Method: **POST**
- URL: http://<baseURL>/v1/users/locale
- Header:

Name	Value
Accept	application/xml
Content-Type	application/xml
X-SAP-LogonToken	The logon token value.

- Body: Paste the template copied from the server response of GET method, specify values for the productlocale, timezone and preferredviewinglocale. For example:

```
<entry xmlns="http://www.w3.org/2005/Atom">
  <content type="application/xml">
    <attrs xmlns="http://www.sap.com/rws/bip">
    <attr name="productlocale" type="string">1033</attr>
    <attr name="timezone" type="string">46008</attr>
    <attr name="preferredviewinglocale" type="string">en_US</attr>
    </attrs>
 </content>
</entry>
```

> **Note**
>
> To pass a valid value for time-zone and locale, view the available time-zones and locales. For more information, see the related topic of this chapter.

Select *Send*. You receive this kind of a response:

```
<entry xmlns="http://www.w3.org/2005/Atom">
   <content type="application/xml">
     <attrs xmlns="http://www.sap.com/rws/bip">
     <attr name="result" type="string">success</attr>
     </attrs>
   </content>
</entry
```

Related Information

Listing Available Locales and Timezones [page 155]

7.6.5 Listing Available Locales and Timezones

GET `http://<baseURL>/v1/users/locale/availabletimezoneandlocale`

Request:

- Method: **GET**
- URL: `http://<baseURL>/v1/users/locale/availabletimezoneandlocale`
  Replace `<baseURL>` with the base URL for RESTful web service requests (such as `10.10.10.10:<port>/biprws`).
- Header:

Name	Value
`Accept`	`application/xml`
`Content-Type`	`application/xml`
`X-SAP-LogonToken`	The logon token value.

- Body: none

Response:

- Header:

Name	Value
`Accept`	`application/xml`
`Content-Type`	`application/xml`
`X-SAP-LogonToken`	The logon token value.

- Body: All available time zones and locales are returned as shown below:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <title type="text">Timezones</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="displayname" type="string">(GMT, No DST) Casablanca,
Monrovia</attr>
                <attr name="title" type="string">time zone</attr>
                <attr name="value" type="string">46000</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">Timezones</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="displayname" type="string">(GMT, DST) Greenwich Mean
Time : Dublin, Edinburgh, Lisbon, London</attr>
                <attr name="title" type="string">time zone</attr>
                <attr name="value" type="string">46002</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">Timezones</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="displayname" type="string">(GMT+01:00, DST)
Amsterdam, Berlin, Bern, Rome, Stockholm, Vienna</attr>
                <attr name="title" type="string">time zone</attr>
                <attr name="value" type="string">46004</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">Timezones</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="displayname" type="string">(GMT+01:00, DST)
Belgrade, Bratislava, Budapest, Ljubljana, Prague</attr>
                <attr name="title" type="string">time zone</attr>
                <attr name="value" type="string">46006</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">preferred viewing locale</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="displayname" type="string">Afrikaans (South Africa)</
attr>
                <attr name="title" type="string">preferred viewing locale</attr>
                <attr name="value" type="string">af_ZA</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">preferred viewing locale</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="displayname" type="string">Albanian (Albania)</attr>
                <attr name="title" type="string">preferred viewing locale</attr>
                <attr name="value" type="string">sq_AL</attr>
            </attrs>
        </content>
    </entry>
```

```
    <entry>
        <title type="text">preferred viewing locale</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="displayname" type="string">Danish (Denmark)</attr>
                <attr name="title" type="string">preferred viewing locale</attr>
                <attr name="value" type="string">da_DK</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.6.6 Changing Password of Current User

GET `http://<baseURL>/v1/users/password`

Request:

- Method: **GET**
- URL: `http://<baseURL>/v1/users/password`
  Replace `<baseURL>` with the base URL for RESTful web service requests (such as `10.10.10.10:<port>/biprws`).
- Header:

Name	Value
`Accept`	`application/xml`
`Content-Type`	`application/xml`
`X-SAP-LogonToken`	The logon token value.

- Body: none

Response:

- Header:

Name	Value
`Accept`	`application/xml`
`Content-Type`	`application/xml`
`X-SAP-LogonToken`	The logon token value.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="oldpassword" type="string"></attr>
            <attr name="newpassword" type="string"></attr>
        </attrs>
```

```
    </content>
</entry>
```

> **Note**
>
> **Copy** the above template.

POST `http://<baseURL>/v1/users/password`

Request:

- Method: **POST**
- URL: `http://<baseURL>/v1/users/password`
- Header:

Name	Value
`Accept`	`application/xml`
`Content-Type`	`application/xml`
`X-SAP-LogonToken`	The logon token value.

- Body: Paste the contents copied in step 1, and specify values for the `oldpassword`(user's current password) and `newpassword` (intended new password). For example:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="oldpassword" type="string">myabcpwd123</attr>
            <attr name="newpassword" type="string">myxyzpwd456</attr>
        </attrs>
    </content>
</entry>
```

Response:

```
<entry xmlns="http://www.w3.org/2005/Atom">
   <content type="application/xml">
     <attrs xmlns="http://www.sap.com/rws/bip">
     <attr name="result" type="string">success</attr>
     </attrs>
   </content>
</entry
```

7.6.7 Changing password of ABAP User

You can now change the SAP ABAP password from the BI platform when

- the password gets expired,
- when an ABAP system administrator changes the password, or
- when you logon for the first time.

> **Note**
>
> A user must exist in an ABAP system and must have access to the BI platform with the same credentials.

This topic explains how you can change your ABAP password through RESTful Web Services.

GET

Request:

- Method: GET
- URL: http://<baseURL>/v1/users/password

> **Note**
>
> Replace <baseURL> with the base URL for RESTful web service requests (such as 10.10.10.10:<port>/biprws).

- Header:

Name	Value
Accept	application/xml
Content-Type	application/xml

- Body: None

Response:

- Header:

Name	Value
Accept	application/xml
Content-Type	application/xml

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
<content type="application/xml">
<attrs xmlns="http://www.sap.com/rws/bip">
<attr name="auth" type="string"></attr>
<attr name="cms" type="string"></attr>
<attr name="username" type="string"></attr>
<attr name="oldpassword" type="string"></attr>
<attr name="newpassword" type="string"></attr>
</attrs>
</content>
</entry>
```

POST

Request:

- Method: **POST**
- URL: `http://<baseURL>/v1/users/password`
- Header:

Name	Value
`Accept`	`application/xml`
`Content-Type`	`application/xml`

- Body: You can use the same XML code template given in the Response of GET operation and specify the values for the `oldpassword` (your current password) and the `newpassword` (your new password). For example:

```
<entry xmlns="http://www.w3.org/2005/Atom">
<content type="application/xml">
<attrs xmlns="http://www.sap.com/rws/bip">
<attr name="auth" type="string"></attr>
<attr name="cms" type="string"></attr>
<attr name="oldpassword" type="string">myabcpwd123</attr>
<attr name="newpassword" type="string">myxyzpwd456</attr>
<attr name="username" type="string"></attr>
</attrs>
</content>
</entry>
```

Response:

```
<entry xmlns="http://www.w3.org/2005/Atom">
<content type="application/xml">
<attrs xmlns="http://www.sap.com/rws/bip">
<attr name="result" type="string">success</attr>
</attrs>
</content>
</entry
>
```

7.7 Deleting Users

DELETE http://<baseURL>/v1/users/<user_id>

Delete a user using the `DELETE` method.

Request:

- Method: DELETE

- URL: `DELETE http://<baseURL>/v1/users/<user_id>`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
Content-Type	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  An XML feed displays success or failure of user delete operation. This example shows the deletion of a user for the resource with ID=6112.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="status" type="string">success</attr>
        </attrs>
    </content>
</entry>
```

7.8 Listing Users

GET http://<baseURL>/v1/users

Get a list of users using the `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/users`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
Content-Type	application/xml
Accept	application/xml
X-SAP-LogonToken	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  An XML feed of all users created in BOE system is displayed.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">user</title>
    <updated>2016-05-27T05:52:12.975Z</updated>
    <link href="http://localhost:6405/biprws/v1/users?page=1&amp;pagesize=50"
rel="self" />
    <link href="http://localhost:6405/biprws/v1/users?page=1&amp;pagesize=50"
rel="first" />
    <link href="http://localhost:6405/biprws/v1/users?page=1&amp;pagesize=50"
rel="last" />
    <entry>
        <title type="text">Administrator</title>
        <updated>2016-05-27T05:51:46.045Z</updated>
        <link href="http://localhost:6405/biprws/v1/users/12" rel="alternate" />
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AfRWaT5_131NlLLf5bRMLKY</attr>
                <attr name="name" type="string">Administrator</attr>
                <attr name="description" type="string">Administrator account</
attr>
                <attr name="id" type="string">12</attr>
                <attr name="fullname" type="string"></attr>
                <attr name="type" type="string">User</attr>
                <attr name="ownerid" type="string">12</attr>
                <attr name="updated" type="string">Fri May 27 07:51:46 CEST
2016</attr>
                <attr name="parentid" type="string">19</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">administrator44</title>
        <updated>2016-03-16T13:10:26.272Z</updated>
        <link href="http://localhost:6405/biprws/v1/users/6112"
rel="alternate" />
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AYEHt8gylJNHtAmzRrmr1cs</attr>
```

```
                <attr name="name" type="string">administrator44</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">6112</attr>
                <attr name="fullname" type="string"></attr>
                <attr name="type" type="string">User</attr>
                <attr name="ownerid" type="string">12</attr>
                <attr name="updated" type="string">Wed Mar 16 14:10:26 CET 2016</
attr>
                <attr name="parentid" type="string">19</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

Sorting users

Sorting is the process of arranging the users in a systematic order.

You can now sort users sequentially in ascending or descending order based on the following attributes:

- Name
- Description
- Id of the user

Filtering users

You can filter users based on the *updated* attribute.

The query parameters for sort and filter is as follows:

Query Parameter

Parameter Name	Parameter Value
Sort	`sort=<+/-><attribute>` **Note** • For ascending order the value is + • For descending order the value is - • By default the users are sorted in ascending based on name of the users **Example** sort=+name

Parameter Name	Parameter Value
Filter	`updated=<Start date year-month-dateThh:mm:ss.Timezone> , <End date year-month-dateThh:mm:ss.Timezone>` **Note** If end date is not specified, by default current date will be considered. **Example** updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z

Filter	• **Exact Search**
	`name=<name>`
	This attribute enables you to search the users based on the name.
	• **Partial Search**
	`name=<name%>`
	This attribute enables you to search the users with partial name.
	○ Placing the % after the name attribute enables you to perform search to find the users which starts with speicifc name.
	
	Example
	`name=user%`
	This displays all the user group that starts with the name user
	
	○ Placing the % before the name attribute enables you to perform search to find the users which ends with speicifc name.
	
	Example
	`name=%user`
	This displays all the user group that ends with the name user
	
	Note
	• In addition, you can filter based on the full name and parentd Id attribute.
	• Partial search support is not there for parent Id attribute.

Parameter Name	Parameter Value
Paging	• `page=<page number>` • `pagesize=<number>` `pagesize` is the number of objects displayed in response **Note** By default `page=1` and `pagesize=50` that is, in one page number 1, the top 50 objects will be displayed in response.

URL: `GET http://<baseURL>/v1/users?sort=<name>&updated=<Start year-month-dateThh:mm:ss.727Z> , <End year-month-dateThh:mm:ss.727Z>&page=<page number>&pagesize=<number>`

Example

URL: `GET http://<baseURL>/v1/users?sort=+name&updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z&page=2&pagesize=5`

once you send the request, the response displays the second page which has five objects and the list of users updated between the specified date and sorted in ascending order.

7.9 Managing User Groups

This section provides you information on RESTful APIs to manage User Group. Using these APIs, you can perform the following:

- Listing user groups
- Creating user group
- Getting user group details
- Listing all users in a user group
- Adding users to a user group
- Removing users from a user Group
- Listing user groups in a user group
- Adding user groups to a user group
- Removing user groups from a user Group
- Modifying user group details
- Deleting user group

7.9.1 Listing User Group

GET http://<baseURL>/v1/usergroups

Get the list of users using the `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/usergroups`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  An XML feed of all users created in BI Platform.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <id>tag:sap.com,2010:bip-rs/infostore</id>
    <title type="text">usergroup</title>
    <updated>2016-05-24T11:26:39.406Z</updated>
    <link href="http://localhost:6405/biprws/v1/usergroups?
page=1&amp;pagesize=50" rel="self"/>
    <link href="http://localhost:6405/biprws/v1/usergroups?
page=1&amp;pagesize=50" rel="first"/>
    <link href="http://localhost:6405/biprws/v1/usergroups?
page=1&amp;pagesize=50" rel="last"/>
    <entry>
        <title type="text">Universe Designer Users</title>
        <id>tag:sap.com,2010:bip-rs/ARWyS4Wb2HVNsW0QTUg2ozs</id>
        <updated>2016-05-17T07:09:30.015Z</updated>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
```

```
                <attr name="cuid" type="string">ARWyS4Wb2HVNsW0QTUg2ozs</attr>
                <attr name="keywords" type="string"></attr>
                <attr name="name" type="string">Universe Designer Users</attr>
                <attr name="description" type="string">Users who can design
universes</attr>
                <attr name="id" type="string">3828</attr>
                <attr name="updated" type="string">Tue May 17 00:09:30 PDT 2016</
attr>
                <attr name="parentid" type="string">20</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">Translators</title>
        <id>tag:sap.com,2010:bip-rs/Ac.U0u28WExKqX7KvRv7tRA</id>
        <updated>2016-05-17T07:09:29.296Z</updated>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">Ac.U0u28WExKqX7KvRv7tRA</attr>
                <attr name="keywords" type="string"></attr>
                <attr name="name" type="string">Translators</attr>
                <attr name="description" type="string">Users granted access to
the Translation Manager application</attr>
                <attr name="id" type="string">3820</attr>
                <attr name="updated" type="string">Tue May 17 00:09:29 PDT 2016</
attr>
                <attr name="parentid" type="string">20</attr>
            </attrs>
        </content>
    </entry>
</feed>
```


Sorting

Sorting is the process of arranging the user groups in a systematic order.

You can now sort user groups sequentially in ascending or descending order based on the following attributes:

- Name
- Description
- Id of the user group

Filtering

You can filter user groups based on the *updated* and *name* attributes.

The query parameters for sort and filter is as follows:

Query Parameter

Parameter Name	Parameter Value
Sort	`sort=<+/-><attribute>` **Note** • For ascending order, the value is + • For descending order, the value is - • By default the user groups are sorted in ascending based on name of the user groups **Example** sort=+name
Filter	`updated=<Start date year-month-dateThh:mm:ss.Timezone> , <End date year-month-dateThh:mm:ss.Timezone>` **Note** If end date is not specified, by default the current date will be taken. **Example** updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z

Parameter Name	Parameter Value

Filter	• **Exact Search**
	`name=<name>`
	This attribute enables you to search the user group based on the name.
	• **Partial Search**
	`name=<name%>`
	This attribute enables you to search the user group with partial name.
	○ Placing the % after the name attribute enables you to perform a search for user groups starting with specific name.
	Example
	`name=user%`
	This displays all the user group that starts with the name user
	○ Placing the % before the name attribute enables you to perform a search for user groups ending with speicifc name.
	Example
	`name=%user`
	This displays all the user group that end with the name user

Paging	• `page=<page number>` • `pagesize=<number>` `pagesize` is the number of objects displayed in the response **Note** By default `page=1` and `pagesize=50` that is, in one page number 1, the top 50 objects will be displayed in response.

URL: `GET http://<baseURL>/v1/usergroups?sort=<name>&updated=<Start year-month-dateThh:mm:ss.727Z> , <End year-month-dateThh:mm:ss.727Z>&page=<page number>&pagesize=<number>`

Example

URL: `GET http://<baseURL>/v1/usergroups?sort=+name&updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z&page=2&pagesize=5`

Once you send the request, the response displays the second page which has five objects and the list of updated between the specified date and sorted in ascending order.

7.9.2 Creating New User Group

Create a user group in BI platform through making request to the `/v1/userGroups/userGroup` URL.

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to create a user group on BI platform.

GET http://<baseURL>/v1/usergroups/usergroup

Make a `GET` request to `/usergroups/usergroup` to receive a template that can be used in the request body of the `POST` method to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/usergroups/usergroup`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
```

```
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="name" type="string"></attr>
            <attr name="description" type="string"></attr>
        </attrs>
    </content>
</entry>
```

POST http://<baseURL>/v1/usergroups/usergroup

To create a user group, use `POST` request to `/v1/usergroups/usergroup` URL.

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/usergroups/usergroup`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="name" type="string">User_Group_Test</attr>
            <attr name="description" type="string"></attr>
        </attrs>
    </content>
</entry>
```

  - Use `<attr name="name" type="string">New_UserGroup_Name</attr>` to define user group name.
  - Use `<attr name="description" type="string">Description</attr>` to define user group description.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
X-SAP-LogonToken	A logon token.
Date	Date and time of the response
Content-Type	Type of content in the response body.

Attribute	Value
Content-Length	Length of content in the response body.

- Body:
  The response body contains the details of user group created .

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <updated>2016-05-24T11:31:28.298Z</updated>
    <link href="http://localhost:6405/biprws/v1/usergroups/usergroup"
rel="alternate"/>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">AcvtoqiL325OheDtowQq6b8</attr>
            <attr name="name" type="string">User_Group-Test</attr>
            <attr name="description" type="string"></attr>
            <attr name="id" type="string">7601</attr>
            <attr name="parentid" type="string">20</attr>
        </attrs>
    </content>
</entry>
```

7.9.3 Getting User Group Details

GET http://<baseURL>/v1/usergroups/<usergroup_id>

Get user group details using `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/usergroups/<usergroup_id>`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
Content-Type	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  An XML feed of user group details in BOE system. This example shows user details for the resource with ID=3828.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <id>tag:sap.com,2010:bip-rs/infostore</id>
    <title type="text">usergroup</title>
    <updated>2016-05-24T11:26:39.406Z</updated>
    <link href="http://10.160.205.83:6405/biprws/v1/usergroups?
page=1&amp;pagesize=50" rel="self"/>
    <link href="http://10.160.205.83:6405/biprws/v1/usergroups?
page=1&amp;pagesize=50" rel="first"/>
    <link href="http://10.160.205.83:6405/biprws/v1/usergroups?
page=1&amp;pagesize=50" rel="last"/>
    <entry>
        <title type="text">Universe Designer Users</title>
        <id>tag:sap.com,2010:bip-rs/ARWyS4Wb2HVNsW0QTUg2ozs</id>
        <updated>2016-05-17T07:09:30.015Z</updated>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">ARWyS4Wb2HVNsW0QTUg2ozs</attr>
                <attr name="keywords" type="string"></attr>
                <attr name="name" type="string">Universe Designer Users</attr>
                <attr name="description" type="string">Users who can design
universes</attr>
                <attr name="id" type="string">3828</attr>
                <attr name="updated" type="string">Tue May 17 00:09:30 PDT 2016</
attr>
                <attr name="parentid" type="string">20</attr>
            </attrs>
        </content>
    </entry>
    <entry>
</feed>
```

7.9.4 Listing All Users in an User Group

GET http://<baseURL>/v1/userGroups/<usergroup_id>/users

Get user details from a user group using `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/userGroups/<usergroup_id>/users>`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
Content-Type	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  An XML feed of user details from the user group on BOE system is displayed. This example shows user details for the resource with ID=11.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">usergroup</title>
    <updated>2016-05-24T11:44:08.818Z</updated>
    <entry>
        <title type="text">Guest</title>
        <updated>2016-05-24T11:44:08.818Z</updated>
        <link href="http://localhost:6405/biprws/v1/usergroups/7602/users"
rel="alternate"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AcgOFGfhCzJEg.VjnPaidmI</attr>
                <attr name="keywords" type="string"></attr>
                <attr name="created" type="string"></attr>
                <attr name="name" type="string">Guest</attr>
                <attr name="description" type="string">Guest account</attr>
                <attr name="id" type="string">11</attr>
                <attr name="fullname" type="string"></attr>
                <attr name="updated" type="string" null="true"/>
                <attr name="email" type="string"></attr>
            </attrs>
        </content>
    </entry>
</feed>
```

Sorting

Sorting is the process of arranging the users in a systematic order.

You can now sort users in a user group sequentially in ascending or descending order based on the following attributes:

- Name
- Description
- Id of the user group

Filtering

You can filter users in a user group based on the *updated* and *name* attribute.

The query parameters for sort and filter is as follows:

Query Parameter

Parameter Name	Parameter Value
Sort	`sort=<+/-><attribute>` **Note** - For ascending order the value is + - For descending order the value is - - By default the users are sorted in ascending based on name of the user group **Example** sort=+name
Filter	`updated=<Start date year-month-dateThh:mm:ss.Timezone> , <End date year-month-dateThh:mm:ss.Timezone>` **Note** If end date is not specified, by default current date will be considered. **Example** updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z

Parameter Name	Parameter Value

Filter	• **Exact Search**
	`name=<name>`
	This attribute enables you to search the user based on the name.
	• **Partial Search**
	`name=<name%>`
	This attribute enables you to search the user with partial name.
	○ Placing the % after the name attribute enables you to perform search to find the user group which starts with speicifc name.
	Example
	`name=user%`
	This displays all the user group that starts with the name user
	○ Placing the % before the name attribute enables you to perform search to find the user group which ends with speicifc name.
	Example
	`name=%user`
	This displays all the user group that ends with the name user

Paging	• `page=<page number>` • `pagesize=<number>` `pagesize` is the number of users displayed in response **Note** By default `page=1` and `pagesize=50` that is, in one page number 1, the top 50 users will be displayed in response.

URL: `GET http://<baseURL>/v1/users?sort=<name>&updated=<Start year-month-dateThh:mm:ss.727Z> , <End year-month-dateThh:mm:ss.727Z>&page=<page number>&pagesize=<number>`

Example

URL: `GET http://<baseURL>/v1/users?sort=+name&updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z&page=2&pagesize=5`

once you send the request, the response displays the second page which has five users and the list of updated between the specified date and sorted in ascending order.

7.9.5 Adding Users to a User Group

PUT http://<baseURL>v1/usergroups/<usergroup_id>/users

To add users to a user group, use `PUT` request to `/v1/usergroups/<usergroup_id>/users>` URL.

Request:

- Method: `PUT`
- URL: `http://<baseURL>/v1/usergroups/<usergroup_id>/users>`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">usergroup</title>
    <updated>2016-05-26T08:57:09.373Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="id" type="string">11</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

  - Use `<attr name="id" type="string">Id</attr>`to define Id of the user.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
X-SAP-LogonToken	A logon token.
Date	Date and time of the response
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:

The response body contains the details of the user added to user group. This example shows user details for the resource with Id=11.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">usergroup</title>
    <updated>2016-05-26T08:58:10.173Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AcgOFGfhCzJEg.VjnPaidmI</attr>
                <attr name="name" type="string">Guest</attr>
                <attr name="id" type="string">11</attr>
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.9.6 Removing Users from a User Group

DELETE http://<baseURL>/v1/usergroups/<usergroup_id>/users

Delete users from a user group using `DELETE` method.

Request:

- Method: DELETE
- URL: `DELETE http://<baseURL>/v1/usergroups/<usergroup_id>/users>`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
Content-Type	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">usergroup</title>
    <updated>2016-05-26T07:53:44.036Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="id" type="string">11</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

- Use `<attr name="id" type="string">Id</attr>`to define Id of the user.

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  An XML feed displays the success or failure message for user delete operation. This example shows the deletion of a user for the resource with Id=11.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">usergroup</title>
    <updated>2016-05-26T09:02:22.569Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AcgOFGfhCzJEg.VjnPaidmI</attr>
                <attr name="name" type="string">Guest</attr>
                <attr name="id" type="string">11</attr>
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.9.7 Listing User Groups in a User Group

GET http://<baseURL>/v1/usergroups/<usergroup_id>/usergroups

List user group details from a user group using `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/usergroups/<usergroup_id>/usergroups`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
Content-Type	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  An XML feed of user group details from the user group on BOE system is displayed. This example shows user details for the resource with ID=6891.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">usergroup</title>
    <updated>2016-05-24T11:57:00.948Z</updated>
    <entry>
        <title type="text">demo2</title>
        <updated>2016-05-24T11:57:00.952Z</updated>
        <link href="http://localhost:6405/biprws/v1/usergroups/6890/usergroups"
rel="alternate"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AfnSG6s.PZVKg6OmZ3msuZA</attr>
                <attr name="keywords" type="string"></attr>
                <attr name="created" type="string"></attr>
                <attr name="name" type="string">demo2</attr>
                <attr name="description" type="string">demo group</attr>
                <attr name="id" type="string">6891</attr>
                <attr name="updated" type="string" null="true"/>
                <attr name="parentid" type="string">20</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

Sorting

Sorting is the process of arranging the user groups in a systematic order.

You can now sort user groups in a user group sequentially in ascending or descending order based on the following attributes:

- Name
- Description
- Id of the user group

Filtering

You can filter user groups in a user group based on the *updated* and *name* attribute.

The query parameters for sort and filter is as follows:

Query Parameter

Parameter Name	Parameter Value
Sort	`sort=<+/-><attribute>` **Note** • For ascending order the value is + • For descending order the value is - • By default the user groups are sorted in ascending based on name of the user group **Example** sort=+name
Filter	`updated=<Start date year-month-dateThh:mm:ss.Timezone> , <End date year-month-dateThh:mm:ss.Timezone>` **Note** If end date is not specified, by default current date will be considered. **Example** updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z

Parameter Name	Parameter Value

Filter	• **Exact Search**
	`name=<name>`
	This attribute enables you to search the user group based on the name.
	• **Partial Search**
	`name=<name%>`
	This attribute enables you to search the user group with partial name.
	○ Placing the % after the name attribute enables you to perform search to find the user group which starts with speicifc name.
	Example
	`name=user%`
	This displays all the user group that starts with the name user
	○ Placing the % before the name attribute enables you to perform search to find the user group which ends with speicifc name.
	Example
	`name=%user`
	This displays all the user group that ends with the name user

Paging	• `page=<page number>` • `pagesize=<number>` `pagesize` is the number of user groups displayed in response **Note** By default `page=1` and `pagesize=50` that is, in one page number 1, the top 50 user groups will be displayed in response.

URL: `GET http://<baseURL>/v1/usergroups?sort=<name>&updated=<Start year-month-dateThh:mm:ss.727Z> , <End year-month-dateThh:mm:ss.727Z>&page=<page number>&pagesize=<number>`

Example

URL: `GET http://<baseURL>/v1/usergroups?sort=+name&updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z&page=2&pagesize=5`

once you send the request, the response displays the second page which has five user groups and the list of updated between the specified date and sorted in ascending order.

7.9.8 Adding User Groups to a User Group

PUT http://<baseURL>v1/usergroups/<usergroup_id>/usergroups

To add user groups to a user group, use `PUT` request to `/v1/usergroups/<usergroup_id>/usergroups>` URL.

Request:

- Method: `PUT`
- URL: `http://<baseURL>/v1/usergroups/<usergroup_id>/users>`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">demo</title>
    <updated>2016-05-26T08:46:31.800Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="id" type="string">7602</attr>
        </attrs>
    </content>
</entry>
</feed>
```

  - Use `<attr name="id" type="string">Id</attr>`to define Id of the user.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.

Attribute	Value
X-SAP-LogonToken	A logon token.
Date	Date and time of the response
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  The response body contains the details of the user added to user group. This example shows user details for the resource with Id=11.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">usergroup</title>
    <updated>2016-05-26T08:52:01.776Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AcMbqXDAS5dIq7fF9929bpw</attr>
                <attr name="name" type="string">demo</attr>
                <attr name="id" type="string">7602</attr>
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.9.9 Removing User Groups from a User Group

DELETE http://<baseURL>/v1/usergroups/<usergroup_id>/usergroups

Delete user groups from a user group using `DELETE` method.

Request:

- Method: DELETE
- URL: `DELETE http://<baseURL>/v1/usergroups/<usergroup_id>/usergroups>`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
Content-Type	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
```

```
    <title type="text">usergroup</title>
    <updated>2016-05-26T07:53:44.036Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="id" type="string">7602</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

- Use `<attr name="id" type="string">Id</attr>`to define Id of the user.

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  An XML feed displays the success or failure message for user group delete operation. This example shows the deletion of a user for the resource with Id=7602.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">usergroup</title>
    <updated>2016-05-26T09:08:57.130Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AcMbqXDAS5dIq7fF9929bpw</attr>
                <attr name="name" type="string">demo</attr>
                <attr name="id" type="string">7602</antr>
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.9.10 Modifying User Group Details

PUT http://<baseURL>v1/usergroups/<usergroup_id>

To modify a user group, use `PUT` request to `/v1/usergroups/<usergroup_id>` URL.

Request:

- Method: `PUT`
- URL: `http://<baseURL>/v1/usergroups/<usergroup_id>`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">AcvtoqiL325OheDtowQq6b8</attr>
            <attr name="keywords" type="string"></attr>
            <attr name="parentcuid" type="string">AW7cVzZFpgFGjOVUdCiOpfE</attr>
            <attr name="created" type="datetime">2016-05-24T11:31:28.256Z</attr>
            <attr name="name" type="string">User_Group_Modified</attr>
            <attr name="description" type="string"></attr>
            <attr name="id" type="string">7601</attr>
            <attr name="updated" type="string" null="true"/>
            <attr name="parentid" type="string">20</attr>
        </attrs>
    </content>
</entry>
```

  - Use `<attr name="cuid" type="string">AdTJAUFCbPNEnbgKiPmAsOE</attr>` to define CU_ID of the user group. .
  - Use `<attr name="keywords" type="string"></attr>` to define keyword for the user group.
  - Use `<attr name="parentcuid" type="string">AdTJAUFCbPNEnbgKiPmAsOE</attr>` to define parent_CUId of the user group. .
  - Use `<attr name="name" type="string">TestUserGroup</attr>` to define title of user group.
  - Use `<attr name="description" type="string">Description</attr>` Description of user group.
  - Use `<attr name="id" type="string">32081</attr>` Id of the user group.
  - Use `<attr name="parentId" type="string">20</attr>` Parent Id of user group.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
X-SAP-LogonToken	A logon token.
Date	Date and time of the response
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:

The response body contains the details of modified user group.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <updated>2016-05-24T11:34:38.367Z</updated>
    <link href="http://localhost:6405/biprws/v1/usergroups/7601"
rel="alternate"/>
    <link href="http://localhost:6405/biprws/v1/usergroups/7601/users"
rel="alternate"/>
    <link href="http://localhost:6405/biprws/v1/usergroups/7601/usergroups"
rel="alternate"/>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">AcvtoqiL325OheDtowQq6b8</attr>
            <attr name="keywords" type="string"></attr>
            <attr name="parentcuid" type="string">AW7cVzZFpgFGjOVUdCiOpfE</attr>
            <attr name="created" type="datetime">2016-05-24T11:31:28.256Z</attr>
            <attr name="name" type="string">User_Group_Modified</attr>
            <attr name="description" type="string"></attr>
            <attr name="id" type="string">7601</attr>
            <attr name="ownerid" type="string">12</attr>
            <attr name="type" type="string">UserGroup</attr>
            <attr name="updated" type="string">Tue May 24 04:33:20 PDT 2016</
attr>
            <attr name="parentid" type="string">20</attr>
        </attrs>
    </content>
</entry>
```

> **Note**
>
> There is no technical limitation on the data length for the POST method. However, in case you want to create large number of users, we recommended you to batch the request in chunks of 500 users.

7.9.11 Deleting User Group

DELETE http://<baseURL>/v1/usergroups/<usergroup_id>

Delete a user group using `DELETE` method.

Request:

- Method: DELETE
- URL: `DELETE http://<baseURL>/v1/usergroups/<usergroup_id>`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
Content-Type	`application/xml`
`Accept`	`application/xml`

Name	Value
X-SAP-LogonToken	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  An XML feed displays the success or failure message for user group delete operation. This example shows the deletion of a user for the resource with ID=39784.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="status" type="string">success</attr>
        </attrs>
    </content>
</entry>
```

7.10 Managing Publications

This section provides you information on publication API's that you can use for publication objects.

7.10.1 Creating Publication

You can create publication on BI platform using the `/publications/publication` URL.

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to create publication on BI platform.

GET http://<baseURL>/v1/publications/publication

Make a `GET` request to `/v1/publications/publication` to receive a template that can be used as a request body for `POST` method.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/publications/publication`
- Body: none

Response:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="keywords" type="string"></attr>
            <attr name="name" type="string"></attr>
            <attr name="description" type="string"></attr>
            <attr name="parentid" type="string"></attr>
        </attrs>
    </content>
</entry>
```

POST http://<baseURL>/v1/publications/publication

To create publication, use `POST` request to `/v1/publications/publication` URL.

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/publications/publication`
- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="keywords" type="string">Test</attr>
            <attr name="name" type="string">Test_Publication</attr>
            <attr name="description" type="string"></attr>
            <attr name="parentid" type="string">6283</attr>
        </attrs>
    </content>
</entry>
```

> **Note**
>
> In the above request, the attributes "name" and "parentid" are mandatory.

  - Use `<attr name="keywords" type="string">Test</attr>` to define publication keyword.
  - Use `<attr name="name" type="string">Test_Publication</attr>` to define publication name.
  - Use `<attr name="description" type="string">Description</attr>` to define publication description.

- Use `<attr name="parentid" type="int32">6283</attr>` to know publication folder ID.

Response:

- Body:
  The response body contains the details of publications created .

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/infostore/12</uri>
    </author>
    <title type="text">Test_Publication</title>
    <updated>2016-06-15T11:53:14.506Z</updated>
    <link href="http://localhost:6405/biprws/v1/publications/29259"/>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">AU3dK9RKIa5GtKzOlRPbh9U</attr>
            <attr name="keywords" type="string">Test</attr>
            <attr name="name" type="string">Test_Publication</attr>
            <attr name="description" type="string"></attr>
            <attr name="id" type="string">29259</attr>
            <attr name="parentid" type="string">6283</antr>
        </attrs>
    </content>
</entry>
```

7.10.2 Listing Publications

GET http://<baseURL>/v1/publications

Get the list of publications using the `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/publications`
- Body: none

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <id>tag:sap.com,2010:bip-rs/infostore</id>
    <title type="text">Publication</title>
    <updated>2016-06-15T11:45:25.152Z</updated>
    <link href="http://localhost:6405/biprws/v1/publications?
page=1&amp;pagesize=50" rel="self"/>
    <link href="http://localhost:6405/biprws/v1/publications?
page=1&amp;pagesize=50" rel="first"/>
    <link href="http://localhost:6405/biprws/v1/publications?
page=1&amp;pagesize=50" rel="last"/>
    <entry>
        <title type="text">cons</title>
        <id>tag:sap.com,2010:bip-rs/AWYjO3zErThIpV2WKXZ9J78</id>
        <author>
```

```
            <name>Administrator</name>
            <uri>http://localhost:6405/biprws/infostore/12</uri>
        </author>
        <updated>2016-06-10T06:44:25.822Z</updated>
        <link href="http://localhost:6405/biprws/v1/publications/28919"
rel="alternate"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AWYjO3zErThIpV2WKXZ9J78</attr>
                <attr name="keywords" type="string"></attr>
                <attr name="name" type="string">cons</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">28919</attr>
                <attr name="parentid" type="string">6283</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">pubO</title>
        <id>tag:sap.com,2010:bip-rs/AcOetAbwASJJjVkYFMwU6jc</id>
        <author>
            <name>Administrator</name>
            <uri>http://localhost:6405/biprws/infostore/12</uri>
        </author>
        <updated>2016-06-09T19:26:04.041Z</updated>
        <link href="http://localhost:6405/biprws/v1/publications/28676"
rel="alternate"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AcOetAbwASJJjVkYFMwU6jc</attr>
                <attr name="keywords" type="string"></attr>
                <attr name="name" type="string">pubO</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">28676</attr>
                <attr name="parentid" type="string">6283</attr>
            </attrs>
        </content>
    </entry>
</feed>
```


Sorting publications

Sorting is the process of arranging the publications in a systematic order.

You can now sort publications sequentially in ascending or descending order based on the following attributes:

- Name
- Description
- Owner ID
- Parent ID

Filtering publications

You can filter publications based on the *updated* attribute.

The query parameters for sort and filter are as follows:

Query Parameter

Parameter Name	Parameter Value
Sort	`sort=<+/-><attribute>` **Note** • For ascending order, the value is + • For descending order, the value is - • By default, the publication are sorted in ascending based on name of the publication **Example** sort=+name
Filter	`updated=<Start date year-month-dateThh:mm:ss.Timezone> , <End date year-month-dateThh:mm:ss.Timezone>` **Note** If end date is not specified, by default current date will be considered. **Example** updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z

Parameter Name	Parameter Value

Filter	• **Exact Search**
	`name=<name>`
	This attribute enables you to search the publications based on the name.
	• **Partial Search**
	`name=<name%>`
	This attribute enables you to search the publications with partial name.
	○ Placing the % after the name attribute enables you to perform a search for publications which start with a specific name.
	Example
	`name=pub%`
	This displays all publication that start with the name pub
	○ Placing the % before the name attribute enables you to perform a search for publications which end with a specific name.
	Example
	`name=%pub`
	This displays all publications that end with the name pub
	Note
	• In addition, you can filter based on the full name and parentd ID attribute.
	• Partial search support is not available for the parent ID attribute.

Paging	• `page=<page number>` • `pagesize=<number>` `pagesize` is the number of objects displayed in the response **Note** By default `page=1` and `pagesize=50` that is, in one page number 1, the top 50 objects will be displayed in the response.

URL: `GET http://<baseURL>/v1/publications?sort=<name>&updated=<Start year-month-dateThh:mm:ss.727Z> , <End year-month-dateThh:mm:ss.727Z>&page=<page number>&pagesize=<number>`

Example

URL: `GET http://<baseURL>/v1/publications?sort=+name&updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z&page=2&pagesize=5`

Once you send the request, the response displays the second page which has five objects, and the list of publications updated between the specified date and sorted in ascending order.

7.10.3 Adding and Deleting Report Documents

You can add report documents for publication on BI platform using the `/v1/publications/<publication_id>/reportdocuments` URL.

You can pass the following types of report document IDs as values:

- CrystalReport
- Webi
- FullClient
- FullClientAddin
- FullClientTemplate

- Use the `GET` method to retrieve report documents in publication.
- Use the `POST` method to add the reports to publication.

GET http://<baseURL>/v1/publications/<publication_id>/reportdocuments

Make a `GET` request to `/v1/publications/<publication_id>/reportdocuments` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/publications/v1/<publication_id>/reportdocuments`
- Body: none

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T04:02:52.115Z</updated>
</feed>
```

POST http://<baseURL>/v1publications/<publication_id>/reportdocuments

To add report documents, use `POST` request to `/publications/<publication_id>/reportdocuments` URL.

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/publications/<publication_id>/reportdocuments`
- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T04:02:52.115Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AS9ukIRdciZLuUS6ESGVRBg</attr>
                <attr name="name" type="string">Drill Demo</attr>
                <attr name="disablerefreshatruntime" type="bool">false</attr>
                <attr name="documentid" type="string">6148</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

  - Use `<attr name="documentid" type="string">6148</attr>` to add report documents for publication.
  - Use `<attr name="disablerefreshatruntime"></attr>` to add document ids for disabling refresh at run time..

Response:

- Body:
  The response body contains the details of user created .

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T04:08:07.131Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AU8SwOvit8dNsDrBhKcWgog</attr>
                <attr name="name" type="string">Input Controls &amp; Filter
Demo</attr>
                <attr name="disablerefreshatruntime" type="bool" null="true"/>
                <attr name="documentid" type="string">6148</attr>
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

DELETE http://<baseURL>/v1/publications/<publication_id>/reportdocuments

Delete report documents using `http://<baseURL>/v1/publications/<publication_id>/reportdocuments` URL.

Request:

- Method: DELETE

- URL: `DELETE http://<baseURL>/v1/publications/<publication_id>/reportdocuments`
- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T03:57:44.076Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AS9ukIRdciZLuUS6ESGVRBg</attr>
                <attr name="name" type="string">Drill Demo</attr>
                <attr name="disablerefreshatruntime" type="bool">false</attr>
                <attr name="documentid" type="string">6144</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T04:19:51.587Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string" null="true"/>
                <attr name="name" type="string" null="true"/>
                <attr name="disablerefreshatruntime" type="bool" null="true"/>
                <attr name="documentid" type="string">6144</attr>
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.10.4 Adding and Deleting Static Documents

A static document is any kind of non-report document. This can include Excel spreadsheets, images, PDFs, and other kinds of supporting documents. A publication can contain any number of static documents.

You can add static documents for publication on BI platform using the `/v1/publications/<publication_id>/staticdocuments` URL.

- Use the `GET` method to retrieve static documents in publication.
- Use the `POST` method to add the documents to publication.

GET http://<baseURL>/v1/publications/<publication_id>/staticdocuments

Make a `GET` request to `/v1/publications/<publication_id>/staticdocuments` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/publications/<publication_id>/staticdocuments`
- Body: none

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T08:33:12.936Z</updated>
</feed>
```

POST http://<baseURL>/publications/<publication_id>/staticdocuments

To add static documents, use `POST` request to `/v1/publications/<publication_id>/staticdocuments` URL.

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/publications/<publication_id>/staticdocuments`
- Body:

```
<<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T04:02:52.115Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AS9ukIRdciZLuUS6ESGVRBg</attr>
                <attr name="name" type="string">Drill Demo</attr>
                <attr name="documentid" type="string">6148</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

  - Use `<attr name="documentID" type="string"></attr>` to add static documents for publication.

Response:

- Body:
  The response body contains the details of user created .

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T08:49:04.588Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AS9ukIRdciZLuUS6ESGVRBg</attr>
                <attr name="name" type="string">Drill Demo</antr>
                <attr name="documentid" type="string">6148</attr>
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

DELETE http://<baseURL>/v1/publications/<publication_id>/staticdocuments

Delete report documents `http://<baseURL>/v1/publications/<publication_id>/staticdocuments` URL.

Request:

- Method: DELETE
- URL: `DELETE http://<baseURL>/v1/publications/<publication_id>/staticdocuments`
- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T04:02:52.115Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="name" type="string">Test</attr>
                <attr name="documentid" type="string">6148</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T08:55:29.340Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AS9ukIRdciZLuUS6ESGVRBg</attr>
                <attr name="name" type="string">Drill Demo</attr>
                <attr name="documentid" type="string">6148</attr>
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.10.5 Adding and Deleting Enterprise Users

Enterprise recipients are individuals who have access to SAP BusinessObjects Business Intelligence platform and have been granted at least read permission on the publication. These recipients typically receive publications in their personal Inboxes, delivered through the managed destination.

You can add only enterprise user for publication on BI platform using the `/v1/publications/<publication_id>/enterpriseusers` URL.

- Use the `GET` method to retrieve enterprise recipients from publication.
- Use the `POST` method to create publication on BI platform.

GET http://<baseURL>/v1/publications/<publication_id>/enterpriseusers

Make a `GET` request to `/v1/publications/<publication_id>/enterpriseusers` to get the enterprise users from publication.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/publications/<publication_id>/enterpriseusers`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Body: none

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T08:59:38.433Z</updated>
</feed>
```

POST http://<baseURL>/v1/publications/<publication_id>/enterpriseusers

To add enterprise user, use `POST` request to `/publications/<id>/enterpriseusers` URL.

Request:

- Method: `POST`
- URL: `http://<baseURL>/publications/<id>/enterpriseUsers`
- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T09:03:35.153Z</updated>
     <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="name" type="string">Guest</attr>
                <attr name="id" type="string">11</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

  - Use `<attr name="enterpriserecipientid" type="string"></attr>` to define enterprise user id (To add multiple users, enter the user ids as comma seperated value).

Response:

- Body:
  The response body contains the details of user created .

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T09:05:18.341Z</updated>
    <entry>
        <author>
            <name>Administrator</name>
```

```
            <uri>http://localhost:6405/biprws/infostore/12</uri>
        </author>
        <title type="text">irfan</title>
        <updated>2016-06-16T09:05:18.356Z</updated>
        <link href="http://localhost:6405/biprws/v1/publications/6409"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AcgOFGfhCzJEg.VjnPaidmI</attr>
                <attr name="name" type="string">Guest</attr>
                <attr name="id" type="string">11</attr>
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

DELETE http://<baseURL>v1/publications/<publication_id>/enterpriseusers

Delete a user using `http://<baseURL>/publications/<publication_id>/enterpriseusers` URL.

Request:

- Method: DELETE
- URL: `DELETE http://<baseURL>/publications/<publication_id>/enterpriseusers`
- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T09:03:35.153Z</updated>
     <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="name" type="string">Guest</attr>
                <attr name="id" type="string">11</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

- Use `<attr name="enterpriserecipientid" type="string"></attr>` to define enterprise user id (To delete multiple users, enter the user ids as comma seperated value).

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T09:08:09.326Z</updated>
    <entry>
        <author>
            <name>Administrator</name>
            <uri>http://localhost:6405/biprws/infostore/12</uri>
        </author>
        <title type="text">irfan</title>
        <updated>2016-06-16T09:08:09.342Z</updated>
        <link href="http://localhost:6405/biprws/v1/publications/6409"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AcgOFGfhCzJEg.VjnPaidmI</attr>
                <attr name="name" type="string">Guest</attr>
                <attr name="id" type="string">11</attr>
                <attr name="status" type="string">success</attr>
```

```
            </attrs>
        </content>
    </entry>
</feed>
```

7.10.6 Adding and Deleting Enterprise User Groups

You can add only enterprise user groups for publication on BI platform using the `/publications/<publication_id>/enterpriseusergroups` URL.

- Use the `GET` method to retrieve enterprise recipients from publication.
- Use the `POST` method to create publication on BI platform.

GET http://<baseURL>/publications/<publication_id>/enterpriseusergroups

Make a `GET` request to `/publications/<publication_id>/enterpriseusergroups` to get the enterprise user groups from publication.

Request:

- Method: GET
- URL: `http://<baseURL>/publications/<publication_id>/enterpriseusergroups`
- Body: none

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T09:09:56.649Z</updated>
</feed>
```

POST http://<baseURL>/v1/publications/<publication_id>/enterpriseusergroups

To create a user, use `POST` request to `/v1/publications/<publication_id>/enterpriseusers` URL.

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/publications/<publication_id>/enterpriseusergroups`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Body:

  ```
  <feed xmlns="http://www.w3.org/2005/Atom">
      <title type="text">Publication</title>
      <updated>2016-06-16T09:09:56.649Z</updated>
  ```

```
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="parentcuid" type="string">AWiHvq39Xe9FtwJZUWJ31h0</
attr>
                <attr name="name" type="string">Everyone</attr>
                <attr name="id" type="string">1</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

- Use `<attr name="enterpriserecipientid" type="string"></attr>` to define enterprise user group id (To add multiple user groups, enter the user group ids as comma seperated value).

Response:

- Body:
  The response body contains the details of user created .

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T09:13:31.812Z</updated>
    <entry>
        <author>
            <name>Administrator</name>
            <uri>http://localhost:6405/biprws/infostore/12</uri>
        </author>
        <title type="text">irfan</title>
        <updated>2016-06-16T09:13:31.828Z</updated>
        <link href="http://localhost:6405/biprws/v1/publications/6409"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AWiHvq39Xe9FtwJZUWJ31h0</attr>
                <attr name="name" type="string">Everyone</attr>
                <attr name="id" type="string">1</attr>
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

DELETE http://<baseURL>/v1/publications/<publication_id>/enterpriseusergroups

Delete a user group using `http://<baseURL>/v1/publications/<publication_id>/enterpriseusergroups` URL.

Request:

- Method: DELETE
- URL: `DELETE http://<baseURL>/v1/publications/<publication_id>/enterpriseusergroups`
- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T09:09:56.649Z</updated>
    <entry>
        <content type="application/xml">
```

```
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="parentcuid" type="string">AWiHvq39Xe9FtwJZUWJ31h0</
attr>
                <attr name="name" type="string">Everyone</attr>
                <attr name="id" type="string">1</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

- Use `<attr name="enterpriserecipientid" type="string"></attr>` to define enterprise user group id (To delete multiple user groups, enter the user group ids as comma seperated value).

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T09:15:47.891Z</updated>
    <entry>
        <author>
            <name>Administrator</name>
            <uri>http://localhost:6405/biprws/infostore/12</uri>
        </author>
        <title type="text">irfan</title>
        <updated>2016-06-16T09:15:47.907Z</updated>
        <link href="http://localhost:6405/biprws/v1/publications/6409"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AWiHvq39Xe9FtwJZUWJ31h0</attr>
                <attr name="name" type="string">Everyone</attr>
                <attr name="id" type="string">1</attr>
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.10.7 Adding and Deleting Dynamic Recipients

Dynamic recipients are individuals who do not have user accounts in SAP BusinessObjects Business Intelligence platform. These recipients are identified through some external data source, which provides the unique identifier, e-mail address, and profile information (if any) for each recipient. Dynamic recipients typically receive publications through an e-mail, delivered through the SMTP destination.

You can add dynamic recipients for publication in BI platform using the `/v1/publications/<publication_id>/dynamicrecipients` URL.

- Use the `GET` method to retrieve dynamic recipients from publication.
- Use the `PUT` method to create add dynamic recipients on BI platform.

GET http://<baseURL>/v1/publications/<publication_id>/dynamicrecipients

Make a `GET` request to `/v1publications/<publication_id>/dynamicrecipients` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/publications/<publication_id>/dynamicrecipients`
- Body: none

Response:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="query" type="string">Query 1</attr>
            <attr name="name" type="string"></attr>
            <attr name="fullname" type="string"></attr>
            <attr name="providerdocumentcuid"
type="string">AS9ukIRdciZLuUS6ESGVRBg</attr>
            <attr name="providerdocumentid" type="string">6148</attr>
            <attr name="providerdocumentname" type="string">Drill Demo</attr>
            <attr name="email" type="string"></attr>
        </attrs>
    </content>
</entry>
```

PUT http://<baseURL>/v1/publications/<publication_id>/dynamicrecipients

To create a user, use `POST` request to `/v1/publications/<publication_id>/dynamicrecipients` URL.

Request:

- Method: `PUT`
- URL: `http://<baseURL>/v1/publications/<publication_id>/dynamicrecipients`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Body:

  ```
  <entry xmlns="http://www.w3.org/2005/Atom">
      <content type="application/xml">
          <attrs xmlns="http://www.sap.com/rws/bip">
              <attr name="query" type="string">Query 1</attr>
              <attr name="name" type="string">User_Name</attr>
              <attr name="fullname" type="string">User_FullName</attr>
              <attr name="providerdocumentcuid" type="string"></attr>
              <attr name="providerdocumentid" type="string">6148</attr>
              <attr name="providerdocumentname" type="string">Drill Demo</attr>
              <attr name="email" type="string">email_id</attr>
          </attrs>
      </content>
  </entry>
  ```

  - Use `<attr name="query" type="string">Query 1</attr>` to query (default value is Query 1) .
  - Use `<attr name="name" type="string">user_name</attr>` to set unique column name from dynamic recipients' reports.
  - Use `<attr name="fullName" type="string">Country Name</attr>` to set name for dynamic recipients.
  - Use `<attr name="providerdocumentcuid" type="string"></attr>` to dynamic recipients' document cuid.

- Use `<attr name="providerdocumentid" type="string">6148</attr>` to dynamic recipients' document id.
- Use `<attr name="providerdocumentname" type="string">Drill Demo</attr>` to dynamic recipients' document id.
- Use `<attr name="email" type="string">email_id</attr>` to set e-mail column from dynamic recipients' report.

Response:

- Body:
  The response body contains the details of user created .

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-16T09:33:36.788Z</updated>
    <entry>
        <author>
            <name>Administrator</name>
            <uri>http://10.160.206.105:6405/biprws/infostore/12</uri>
        </author>
        <title type="text">irfan</title>
        <updated>2016-06-16T09:33:36.788Z</updated>
        <link href="http://10.160.206.105:6405/biprws/v1/publications/6409"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="query" type="string">Query 1</attr>
                <attr name="name" type="string"></attr>
                <attr name="fullname" type="string"></attr>
                <attr name="providerdocumentcuid" type="string"></attr>
                <attr name="providerdocumentid" type="string">6148</attr>
                <attr name="providerdocumentname" type="string">Drill Demo</attr>
                <attr name="email" type="string"></attr>
            </attrs>
        </content>
    </entry>
</feed>
```

DELETE http://<baseURL>/v1/publications/<publication_id>/dynamicrecipients

Delete a user using `http://<baseURL>/v1/publications/<publication_id>/dynamicrecipients` URL.

Response:

- Method: DELETE
- URL: `DELETE http://<baseURL>/v1/publications/<publication_id>/dynamicrecipients`
- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="query" type="string">Query 1</attr>
            <attr name="name" type="string"></attr>
            <attr name="fullname" type="string"></attr>
            <attr name="providerdocumentcuid" type="string"></attr>
            <attr name="providerdocumentid" type="string">6148</attr>
            <attr name="providerdocumentname" type="string">Drill Demo</attr>
            <attr name="email" type="string"></attr>
```

```
        </attrs>
    </content>
</entry>
```

An XML feed displays success or failure of user delete operation. This example shows the deletion of a user for the resource with id=6148.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.10.8 Configuring Output Format for Reports

You can configure output formats for report documents in publication on BI platform using the `/v1/publications/<publication_id>/outputformats` URL.

The following table shows all supported output formats and the report document kinds details:

Output Format Constant	Crystal Reports	Web Intelligence
CRYSTAL_REPORT	YES	
EXCEL	YES	YES
EXCEL_DATA_ONLY	YES	
MHTML	YES	YES
PDF	YES	YES
RTF	YES	
RTF_EDITABLE	YES	
TEXT_CHARACTER_SEPARATED	YES	
TEXT_PAGINATED	YES	
TEXT_PLAIN	YES	
TEXT_TAB_SEPARATED	YES	
TEXT_TAB_SEPARATED_TEXT	YES	
USER_DEFINED	N/A	N/A

Output Format Constant	Crystal Reports	Web Intelligence
WORD	YES	
XML	YES	

- Use the `GET` method to retrieve enabled output formats of the document.
- Use the `PUT` method to configure output formats.

GET http://<baseURL>/v1/publications/<publication_id>/outputformats

Make a `GET` request to `/v1/publications/<publication_id>/outputformats` to retrieve enabled output formats of the document that can be used in the request body of a `PUT` method.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/publications/<publication_id>/outputformats`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Body: none

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-17T15:09:54.450Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AQtkbbSqN4NOj3ydf.Sw1lY</attr>
                <attr name="name" type="string">Formatting Sample</attr>
                <attr name="id" type="string">5371</attr>
                <attr name="outputformats" type="string">[EXCEL, WEBI]</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

> **Note**
>
> In response you can see the report document ids (comma separated values) in the respective output format tags, if the format is enabled for the same.

PUT http://<baseURL>/v1/publications/<publication_id>/outputformats

To configure output format, use `PUT` request to `/v1/publications/<publication_id>/outputformats` URI.

Request:

- Method: `PUT`
- URL: `http://<baseURL>/v1/publications/<publication_id>/outputformats`
- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-17T15:09:54.450Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AQtkbbSqN4NOj3ydf.Sw1lY</attr>
                <attr name="name" type="string">Formatting Sample</attr>
                <attr name="id" type="string">5371</attr>
                <attr name="outputformats" type="string">[EXCEL, WEBI]</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

  - You must use comma seperated (,) document IDs in respective tags.
  - Each attribute represents an output format.
  - The document formats against which the report document IDs are entered is only retained in the server, other formats are removed.

Response:

- Body:
  The response body contains:
  - Success and failure list
  - Failure details of each output format

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <updated>2016-06-17T15:12:19.866Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AQtkbbSqN4NOj3ydf.Sw1lY</attr>
                <attr name="name" type="string">Formatting Sample</attr>
                <attr name="failed" type="string"></attr>
                <attr name="id" type="string">5371</antr>
                <attr name="outputformats" type="string">[EXCEL,WEBI]</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.10.9 Configuring Destination Forms

GET http://<baseURL>/v1/publications/<publication_id>/destinations

To get all available destinations for publication using the `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/publications/<publication_id>/destinations`
- Body: none

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://10.160.206.105:6405/biprws/infostore/12</uri>
    </author>
    <publication_id>6410</id>
    <title type="text">pub1</title>
</feed>
```

7.10.10 Configuring Inbox Destination Plugin

Configure inbox destination plugin for publication in BI platform using the `/v1/publications/<publication_id>/destinations/inbox` URL.

- Use the `GET` method to retrieve the configuration information if inbox destination is already enabled, else in response you will receive an default template.
- Use the `PUT` method to configure publication on BI platform.

GET http://<baseURL>/v1/publications/<publication_id>/destinations /inbox

Make a `GET` request to `/v1/publications/<publication_id>/destinations/inbox` the configuration information if inbox destination is already enabled, else in response you will receive an default template.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/publications/<publication_id>/destinations/inbox`
- Body: none

Response:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="fileextension" type="bool">true</attr>
            <attr name="zipfile" type="bool">false</attr>
            <attr name="deliverperuser" type="bool">true</attr>
            <attr name="mergeexportedpdf" type="bool">false</attr>
            <attr name="sendascopy" type="bool">true</attr>
            <attr name="targetobjectname" type="string"></attr>
        </attrs>
    </content>
</entry>
```

PUT http://<baseURL>/v1/publications/<publication_id>/destinations /inbox

To configure inbox destination plugin, use `PUT` request to `/v1/publications/<publication_id>/destinations/inbox` URL.

Request:

- Method: `PUT`
- URL: `http://<baseURL>/v1/publications/<publication_id>/destinations/inbox`
- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="fileextension" type="bool">true</attr>
            <attr name="zipfile" type="bool">false</attr>
            <attr name="deliverperuser" type="bool">true</attr>
            <attr name="mergeexportedpdf" type="bool">false</attr>
            <attr name="sendascopy" type="bool">true</antr>
            <attr name="targetobjectname" type="string"></attr>
        </attrs>
    </content>
</entry>
```

  - Use `<attr name="fileextension" type="bool">true</attr>` boolean value to set file extension.
  - Use `<attr name="zipfile" type="bool">false</attr>` boolean value to set zip file.
  - Use `<attr name="deliverperuser" type="bool">true</attr>` boolean value to set deliver per user.
  - Use `<attr name="mergeexportedpdf" type="bool">false</attr>` boolean value to set merge export as pdf.
  - Use `<attr name="sendascopy" type="bool">true</attr>` to set send as a copy or shortcut.
  - Use `<attr name="targetobjectname" type="string"></attr>` to set specific name for publication.

> **Note**
>
> The query parameters against which the parameter values are entered is only retained in the server, other data is removed.

Response:

- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/infostore/12</uri>
    </author>
    <publication_id>6410</id>
    <title type="text">pub1</title>
    <updated>2016-06-17T11:19:34.892Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="fileextension" type="bool">true</attr>
                <attr name="zipfile" type="bool">false</attr>
                <attr name="deliverperuser" type="bool">true</attr>
                <attr name="mergeexportedpdf" type="bool">false</attr>
```

```
                <attr name="sendascopy" type="bool">true</attr>
                <attr name="targetobjectname" type="string"></attr>
            </attrs>
        </content>
    </entry>
</feed>
```

DELETE method

Delete destination plugin using `http://<baseURL>/v1/publications/<publication_id>/inbox` URL.

- Method: `DELETE`
- URL: `http://<baseURL>/v1/publications/<publication_id>/destinations/inbox`
- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="fileextension" type="bool">true</attr>
            <attr name="zipfile" type="bool">false</attr>
            <attr name="deliverperuser" type="bool">true</attr>
            <attr name="mergeexportedpdf" type="bool">false</attr>
            <attr name="sendascopy" type="bool">true</attr>
            <attr name="targetobjectname" type="string"></attr>
        </attrs>
    </content>
</entry>>
```

Response:

An XML feed displays success or failure of delete operation

```
<feed
    xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <entry>
        <content type="application/xml">
            <attrs
                xmlns="http://www.sap.com/rws/bip">
                <attr name="status" type="string">Success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.10.11 Configuring SMTP Destination Plugin

Configure SMTP destination plugin for publication in BI platform using the `/v1/publications/<publication_id>/destinations/smtp` URL.

- Use the `GET` method to retrieve the configuration information if SMTP destination is already enabled, else in response you will receive an default template.

- Use the `PUT` method to configure SMTP publication on BI platform.

GET http://<baseURL>/v1/publications/<publication_id>/destinations/smtp

Make a `GET` request to `/v1/publications/<publication_id>/destinations/smtp` to retrieve the configuration information if SMTP destination is already enabled, else in response you will receive an default template.

Response:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="fileextension" type="bool">true</attr>
            <attr name="zipfile" type="bool">false</attr>
            <attr name="subject" type="string"></attr>
            <attr name="deliverperuser" type="bool">true</attr>
            <attr name="mergeexportedpdf" type="bool">false</attr>
            <attr name="targetobjectname" type="string"></attr>
            <attr name="message" type="string"></attr>
            <attr name="attachmentsenabled" type="bool">true</attr>
            <attr name="enablessl" type="bool">false</attr>
            <attr name="bccaddresses" type="string"></attr>
            <attr name="ccaddresses" type="string"></attr>
            <attr name="senderaddress" type="string"></attr>
            <attr name="toaddresses" type="string"></attr>
        </attrs>
    </content>
</entry>
```

PUT http://<baseURL>/v1/publications/<publication_id>/destinations /smtp

To configure SMTP destination plugin, use `PUT` request to `/v1/publications/<publication_id>/destinations/smtp` URL.

Request:

- Method: `PUT`
- URL: `http://<baseURL>/v1/publications/<publication_id>/destinations/smtp`
- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="fileextension" type="bool">true</attr>
            <attr name="zipfile" type="bool">false</attr>
            <attr name="subject" type="string"></attr>
            <attr name="deliverperuser" type="bool">true</attr>
            <attr name="mergeexportedpdf" type="bool">false</attr>
            <attr name="targetobjectname" type="string"></attr>
            <attr name="message" type="string"></attr>
            <attr name="attachmentsenabled" type="bool">true</antr>
            <attr name="enablessl" type="bool">false</attr>
            <attr name="bccaddresses" type="string"></attr>
            <attr name="ccaddresses" type="string"></attr>
```

```
            <attr name="senderaddress" type="string"></attr>
            <attr name="toaddresses" type="string"></attr>
        </attrs>
    </content>
</entry>
```

- Use `<attr name="fileextension" type="bool">true</attr>` boolean value to set file extension.
- Use `<attr name="zipfile" type="bool">false</attr>` boolean value to set zip file.
- Use `<attr name="subject" type="string"></attr>` to define subject value.
- Use `<attr name="deliverperuser" type="bool">true</attr>` boolean value to set deliver per user.
- Use `<attr name="mergeexportedpdf" type="bool">false</attr>` boolean value to set merge export as pdf.
- Use `<attr name="targetobjectname" type="string"></attr>` to set specific name for publication.
- Use `<attr name="message" type="string"></attr>` to enter the message.
- Use `<attr name="attachmentsenabled" type="bool">true</attr>` boolean value to enable attachments.
- Use `<attr name="enablessl" type="bool">false</attr>` boolean value to enable SSL.
- Use `<attr name="bccaddresses" type="string"></attr>`to define Bcc recipient.
- Use `<attr name="ccaddresses" type="string"></attr>` to define Cc recipient.
- Use `<attr name="toaddresses" type="string"></attr>` to define To recipient.

Note

- The query parameters against which the parameter values are entered is only retained in the server, other data is removed.
- E-mail addresses must be semicolon (;) separated.

Response:

- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/infostore/12</uri>
    </author>
    <publication_id>6410</id>
    <title type="text">pub1</title>
    <updated>2016-06-17T11:51:03.012Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="fileextension" type="bool">true</attr>
                <attr name="zipfile" type="bool">false</attr>
                <attr name="subject" type="string">Test1</attr>
                <attr name="deliverperuser" type="bool">true</attr>
                <attr name="mergeexportedpdf" type="bool">true</attr>
                <attr name="targetobjectname" type="string">SMTP</attr>
                <attr name="message" type="string">SMTP</attr>
                <attr name="attachmentsenabled" type="bool">true</attr>
                <attr name="enablessl" type="bool">false</attr>
                <attr name="bccaddresses" type="string"></attr>
                <attr name="ccaddresses" type="string"></attr>
                <attr name="senderaddress" type="string"></attr>
                <attr name="toaddresses" type="string">xyz@sap.com</attr>
```

```
            </attrs>
        </content>
    </entry>
</feed>
```

DELETE method

Delete destination SMTP using `http://<baseURL>/v1/publications/<publication_id>/destinations/smtp` URL.

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.10.12 Configuring FTP Destination Plugin

Configure FTP destination plugin for publication in BI platform using the `/v1/publications/<publication_id>/destinations/ftp` URL.

- Use the `GET` method to retrieve the configuration information if FTP destination is already enabled, else in response you will receive an default template.
- Use the `PUT` configure FTP destination plugin for publication on BI platform.

GET http://<baseURL>/v1/publications/<publication_id>/destinations/ftp

Make a `GET` request to `/v1/publications/<publication_id>/destinations/ftp` to retrieve the configuration information if FTP destination is already enabled, else in response you will receive an default template.

Response:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="fileextension" type="bool">true</attr>
            <attr name="zipfile" type="bool">false</attr>
            <attr name="password" type="string"></attr>
            <attr name="port" type="int32">21</attr>
            <attr name="deliverperuser" type="bool">true</attr>
            <attr name="mergeexportedpdf" type="bool">false</attr>
```

```
            <attr name="servername" type="string"></attr>
            <attr name="targetobjectname" type="string"></attr>
            <attr name="directory" type="string">.</attr>
            <attr name="account" type="string"></attr>
            <attr name="username" type="string"></attr>
        </attrs>
    </content>
</entry>
```

PUT http://<baseURL>/v1/publications/<publication_id>/destinations/ftp

To configure FTP destination plugin, use `PUT` method request for `/publications/<publication_id>/destinations/ftp` URL.

Request:

- Method: `PUT`
- URL: `http://<baseURL>/v1/publications/<publication_id>/destinations/ftp`
- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="fileextension" type="bool">true</attr>
            <attr name="zipfile" type="bool">false</attr>
            <attr name="password" type="string"></attr>
            <attr name="port" type="int32">21</attr>
            <attr name="deliverperuser" type="bool">true</attr>
            <attr name="mergeexportedpdf" type="bool">false</attr>
            <attr name="servername" type="string"></attr>
            <attr name="targetobjectname" type="string"></attr>
            <attr name="directory" type="string">.</attr>
            <attr name="account" type="string"></attr>
            <attr name="username" type="string">FTP</attr>
        </attrs>
    </content>
</entry>
```

  - The query parameters against which the parameter values are entered is only retained in the server, other data is removed.

Response:

- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/infostore/12</uri>
    </author>
    <publication_id>6410</id>
    <title type="text">pub1</title>
    <updated>2016-06-17T12:11:13.571Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="fileextension" type="bool">true</attr>
                <attr name="zipfile" type="bool">false</attr>
                <attr name="password" type="string"></attr>
                <attr name="port" type="int32">21</attr>
```

```
                <attr name="deliverperuser" type="bool">true</attr>
                <attr name="mergeexportedpdf" type="bool">false</attr>
                <attr name="servername" type="string"></attr>
                <attr name="targetobjectname" type="string"></attr>
                <attr name="directory" type="string">.</attr>
                <attr name="account" type="string"></attr>
                <attr name="username" type="string">FTP</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

- Use `<attr name="fileextension" type="bool">true</attr>` boolean value to set file extension.
- Use `<attr name="zipfile" type="bool">false</attr>` boolean value to set zip file.
- Use `<attr name="password" type="string"></attr>` to define the password.
- Use `<attr name="deliverperuser" type="bool">true</attr>` boolean value to set deliver per user.
- Use `<attr name="port" type="string">21</attr>` to enter port number.
- Use `<attr name="mergeexportedpdf" type="bool">false</attr>` boolean value to set merge export as pdf.
- Use `<attr name="servername" type="string"></attr>` to define server name.
- Use `<attr name="targetobjectname" type="string"></attr>` to set specific name for publication.
- Use `<attr name="directory" type="string">.</attr>` to define the directory.
- Use `<attr name="enablessl" type="bool">false</attr>` boolean value to enable SSL.
- Use `<attr name="account" type="string"></attr>`to define account type.
- Use `<attr name="username" type="string">FTP</attr>` to define user name.

DELETE method

Delete destination FTP using `http://<baseURL>/v1/publications/<publication_id>/destinations/ftp` URL.

- Method: `DELETE`
- URL: `http://<baseURL>/v1/publications/<publication_id>/destinations/ftp`
- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="fileextension" type="bool">true</attr>
            <attr name="zipfile" type="bool">false</attr>
            <attr name="password" type="string"></attr>
            <attr name="port" type="int32">21</attr>
            <attr name="deliverperuser" type="bool">true</attr>
            <attr name="mergeexportedpdf" type="bool">false</attr>
            <attr name="servername" type="string"></attr>
            <attr name="targetobjectname" type="string"></attr>
            <attr name="directory" type="string">.</attr>
            <attr name="account" type="string"></attr>
            <attr name="username" type="string">FTP</attr>
        </attrs>
    </content>
</entry>
```

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.10.13 Configuring SFTP Destination Plugin

Configure SFTP destination plugin for publication in BI platform using the `/v1/publications/<publication_id>/destinations/sftp` URL.

- Use the `GET` method to retrieve the configuration information if SFTP destination is already enabled, else in response you will receive an default template.
- Use the `PUT` configure SFTP destination plugin for publication on BI platform.

GET http://<baseURL>/v1/publications/<publication_id>/destinations / inbox/publications/<publication_id>/destinations/sftp

Make a `GET` request to `/publications/<publication_id>/destinations/sftp` to retrieve the configuration information if SFTP destination is already enabled, else in response you will receive an default template.

Response:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="fileextension" type="bool">true</attr>
            <attr name="zipfile" type="bool">false</attr>
            <attr name="password" type="string"></attr>
            <attr name="port" type="int32">22</attr>
            <attr name="deliverperuser" type="bool">true</attr>
            <attr name="fingerprint" type="string"></attr>
            <attr name="mergeexportedpdf" type="bool">false</attr>
            <attr name="servername" type="string"></attr>
            <attr name="targetobjectname" type="string"></attr>
            <attr name="directory" type="string">.</attr>
            <attr name="account" type="string"></attr>
            <attr name="username" type="string"></attr>
        </attrs>
    </content>
</entry>
```

PUT http://<baseURL>/v1/publications/<publication_id>/destinations/sftp

To configure SFTP destination plugin, use `PUT` method request for `/v1/publications/<publication_id>/destinations/sftp` URL.

Request:

- Method: `PUT`
- URL: `http://<baseURL>/v1/publications/<publication_id>/destinations/sftp`
- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="fileextension" type="bool">true</attr>
            <attr name="zipfile" type="bool">false</attr>
            <attr name="password" type="string"></attr>
            <attr name="port" type="int32">22</attr>
            <attr name="deliverperuser" type="bool">true</attr>
            <attr name="fingerprint" type="string"></attr>
            <attr name="mergeexportedpdf" type="bool">false</attr>
            <attr name="servername" type="string"></attr>
            <attr name="targetobjectname" type="string"></attr>
            <attr name="directory" type="string">.</attr>
            <attr name="account" type="string"></attr>
            <attr name="username" type="string"></attr>
        </attrs>
    </content>
</entry>
```

-
  - Use `<attr name="fileextension" type="bool">true</attr>` boolean value to set file extension.
  - Use `<attr name="zipfile" type="bool">false</attr>` boolean value to set zip file.
  - Use `<attr name="password" type="string"></attr>` to define the password.
  - Use `<attr name="port" type="string">21</attr>` to enter port number.
  - Use `<attr name="deliverperuser" type="bool">true</attr>` boolean value to set deliver per user.
  - Use `<attr name="fingerprint" type="string"></attr>` to define fingerprint.
  - Use `<attr name="mergeexportedpdf" type="bool">false</attr>` boolean value to set merge export as pdf.
  - Use `<attr name="servername" type="string"></attr>` to define server name.
  - Use `<attr name="targetobjectname" type="string"></attr>` to set specific name for publication.
  - Use `<attr name="directory" type="string">.</attr>` to define the directory.
  - Use `<attr name="account" type="string"></attr>`to define account type.
  - Use `<attr name="username" type="string">FTP</attr>` to define user name.

Response:

- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/infostore/12</uri>
    </author>
    <publication_id>6410</id>
```

```
    <title type="text">pub1</title>
    <updated>2016-06-17T12:23:25.682Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="fileextension" type="bool">true</attr>
                <attr name="zipfile" type="bool">false</attr>
                <attr name="password" type="string"></attr>
                <attr name="port" type="int32">22</attr>
                <attr name="deliverperuser" type="bool">true</attr>
                <attr name="fingerprint" type="string"></attr>
                <attr name="mergeexportedpdf" type="bool">false</attr>
                <attr name="servername" type="string"></attr>
                <attr name="targetobjectname" type="string"></attr>
                <attr name="directory" type="string">.</attr>
                <attr name="account" type="string"></attr>
                <attr name="username" type="string">SFTP</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

DELETE method

Delete destination SFTP using `http://<baseURL>/v1/publications/<publication_id>/destinations/sftp` URL.

- Method: `DELETE`
- URL: `http://<baseURL>/v1/publications/<publication_id>/destinations/sftp`
- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="fileextension" type="bool">true</attr>
            <attr name="zipfile" type="bool">false</attr>
            <attr name="deliverperuser" type="bool">true</attr>
            <attr name="mergeexportedpdf" type="bool">false</attr>
            <attr name="sendascopy" type="bool">true</attr>
            <attr name="targetobjectname" type="string"></attr>
        </attrs>
    </content>
</entry>>
```

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.10.14 Configuring File System Destination Plugin

Configure file system destination plugin in for publication BI platform using the `/v1/publications/<publication_id>/destinations/disk` URL.

- Use the `GET` method to retrieve the configuration information if file system destination is already enabled, else in response you will receive an default template.
- Use the `PUT` configure file system destination plugin for publication on BI platform.

GET http://<baseURL>/v1/publications/<publication_id>/publications/<publication_id>/destinations/disk

Make a `GET` request to `/v1/publications/<publication_id>/destinations/disk` to retrieve the configuration information if file system destination is already enabled, else in response you will receive an default template.

Response:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="fileextension" type="bool">true</attr>
            <attr name="zipfile" type="bool">false</attr>
            <attr name="password" type="string"></attr>
            <attr name="mergeexportedpdf" type="bool">false</attr>
            <attr name="deliveperuser" type="bool">true</attr>
            <attr name="targetobjectname" type="string"></attr>
            <attr name="directory" type="string"></attr>
            <attr name="username" type="string"></attr>
        </attrs>
    </content>
</entry>
```

PUT http://<baseURL>/v1/publications/<publication_id>/ destinations /disk

To configure file system destination plugin , use `PUT` request to `/publications/<publication_id>/destinations/disk` URL.

Request:

- Method: `PUT`
- URL: `http://<baseURL>/v1/publications/<publication_id>/destinations/disk`
- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="fileextension" type="bool">true</attr>
            <attr name="zipfile" type="bool">false</attr>
            <attr name="password" type="string"></attr>
```

```
            <attr name="mergeexportedpdf" type="bool">false</attr>
            <attr name="deliveperuser" type="bool">true</attr>
            <attr name="targetobjectname" type="string"></attr>
            <attr name="directory" type="string"></attr>
            <attr name="username" type="string">disk</attr>
        </attrs>
    </content>
</entry>
```

-
  - Use `<attr name="fileextension" type="bool">true</attr>` boolean value to set file extension.
  - Use `<attr name="zipfile" type="bool">false</attr>` boolean value to set zip file.
  - Use `<attr name="password" type="string"></attr>` to define the password.
  - Use `<attr name="fingerprint" type="string"></attr>` to define fingerprint.
  - Use `<attr name="mergeexportedpdf" type="bool">false</attr>` boolean value to set merge export as pdf.
  - Use `<attr name="deliverperuser" type="bool">true</attr>` boolean value to set deliver per user.
  - Use `<attr name="targetobjectname" type="string"></attr>` to set specific name for publication.
  - Use `<attr name="directory" type="string">.</attr>` to define the directory.
  - Use `<attr name="username" type="string">FTP</attr>` to define user name.
-

Response:

- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://10.160.206.105:6405/biprws/infostore/12</uri>
    </author>
    <publication_id>6410</id>
    <title type="text">pub1</title>
    <updated>2016-06-17T13:35:34.181Z</updated>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="fileextension" type="bool">true</attr>
                <attr name="zipfile" type="bool">false</attr>
                <attr name="password" type="string">false</attr>
                <attr name="mergeexportedpdf" type="bool">false</attr>
                <attr name="deliveperuser" type="bool">true</attr>
                <attr name="targetobjectname" type="string"></attr>
                <attr name="directory" type="string"></attr>
                <attr name="username" type="string">disk</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

DELETE method

Delete destination diskUnmanaged from publication using `http://<baseURL>/v1/publications/<publication_id>/destinations/disk` URL.

- Method: `DELETE`
- URL: `http://<baseURL>/v1/publications/<publication_id>/destinations/disk`
- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="fileextension" type="bool">true</attr>
            <attr name="zipfile" type="bool">false</attr>
            <attr name="password" type="string"></attr>
            <attr name="mergeexportedpdf" type="bool">false</attr>
            <attr name="deliveperuser" type="bool">true</attr>
            <attr name="targetobjectname" type="string"></attr>
            <attr name="directory" type="string"></attr>
            <attr name="username" type="string">disk</attr>
        </attrs>
    </content>
</entry>
```

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.10.15 Configuring Target Object Name Per Document

Configure Target Object Name Per Document for publication in BI platform using the `/v1/publications/<publication_id>/destinations/<destination>/targetobjectperdocument` URL.

- Use the `GET` method to retrieve the configuration information if Target Object Name Per Document is already enabled, else in response you will receive an default template.
- Use the `PUT` method to configure Target Object Name Per Document publication on BI platform.

GET http://<baseURL>/v1/publications/<publication_id>/destinations/<destination>/targetobjectperdocument

Make a `GET` request to `/v1/publications/<publication_id>/destinations/<destination>/targetobjectperdocument` to retrieve the configuration information if Target Object Name Per Document publication is already enabled, else in response you will receive an default template.

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
```

```
    <title type="text">pub</title>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AXpyoryP_Z5BoXVoM2_3kWQ</attr>
                <attr name="name" type="string">Fold Unfold Sample</attr>
                <attr name="id" type="string">5489</attr>
                <attr name="specificname" type="string">new2%</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AW4AVT1AUhVAogA6P7OQv9c</attr>
                <attr name="name" type="string">Charting Samples</attr>
                <attr name="id" type="string">5488</antr>
                <attr name="specificname" type="string">new</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

PUT http://<baseURL>/v1/publications/<publication_id>/destinations/<destination>/targetobjectperdocument

To configure Target Object Name Per Document plugin, use `PUT` request to `v1/publications/<publication_id>/destinations/<destination>/targetobjectperdocument` URL.

Request:

- Method: `PUT`
- URL: `http://<baseURL>/v1/publications/<publication_id>/destinations/<destination>/targetobjectperdocument`
- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">pub</title>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="id" type="string">5489</attr>
                <attr name="specificname" type="string">new2%</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="id" type="string">5488</attr>
                <attr name="specificname" type="string">new</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

  - Use `<attr name="id" type="string">5488</attr>` to define the id.
  - Use `<attr name="specificname" type="string">new</attr>` to define the specific name.

> **Note**
>
> - The query parameters against which the parameter values are entered is only retained in the server, other data is removed.
> - E-mail addresses must be semicolon (;) separated.

Response:

- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">pub</title>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AXpyoryP_Z5BoXVoM2_3kWQ</attr>
                <attr name="name" type="string">Fold Unfold Sample</attr>
                <attr name="id" type="string">5489</attr>
                <attr name="specificname" type="string">2.%EXT%</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AW4AVT1AUhVAogA6P7OQv9c</attr>
                <attr name="name" type="string">Charting Samples</antr>
                <attr name="id" type="string">5488</attr>
                <attr name="specificname" type="string">1.%EXT%</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

DELETE method

Delete Target Object Name Per Document using `http://<baseURL>/v1/publications/<publication_id>/destinations/<destination>/targetobjectperdocument` URL.

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">pub</title>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.10.16 Listing Schedule Instances of Publication

GET http://<baseURL>/v1/publications/<publication_id>/schedules

To list all the scheduled instances of publications, use the `GET http://<baseURL>/v1/publications/<publication_id>/schedules` URL.

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Schedules</title>
    <updated>2016-06-17T13:54:06.067Z</updated>
    <link href="http://localhost:6405/biprws/v1/publications/6410/schedules?
page=1&amp;pagesize=50" rel="self"/>
    <link href="http://localhost:6405/biprws/v1/publications/6410/schedules?
page=1&amp;pagesize=50" rel="first"/>
    <link href="http://localhost:6405/biprws/v1/publications/6410/schedules?
page=1&amp;pagesize=50" rel="last"/>
    <entry>
        <title type="text">pub1</title>
        <link href="http://localhost:6405/biprws/v1/publications/6410/schedules/
6883"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="owner" type="string">Administrator</attr>
                <attr name="cuid" type="string">AbpDNk7FosZOh3R60.sfGBQ</attr>
                <attr name="created" type="string">Fri Jun 17 06:53:55 PDT 2016</
attr>
                <attr name="parentcuid" type="string">Af5AnFKOlU9Jj_XM_MrqUkY</attr>
                <attr name="name" type="string">pub1</attr>
                <attr name="id" type="string">6883</attr>
                <attr name="type" type="string">Publication</attr>
                <attr name="ownerid" type="int32">12</attr>
                <attr name="parentid" type="string">6410</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">pub1</title>
        <link href="http://localhost:6405/biprws/v1/publications/6410/schedules/
6885"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="owner" type="string">Administrator</attr>
                <attr name="cuid" type="string">ARXYPatVXRZPpsrbCmiLsY4</attr>
                <attr name="created" type="string">Fri Jun 17 06:53:58 PDT 2016</
attr>
                <attr name="parentcuid" type="string">Af5AnFKOlU9Jj_XM_MrqUkY</attr>
                <attr name="name" type="string">pub1</attr>
                <attr name="id" type="string">6885</attr>
                <attr name="type" type="string">Publication</attr>
                <attr name="ownerid" type="int32">12</attr>
                <attr name="parentid" type="string">6410</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

Schedule Status Table

Schedule Status Code	Description
0	Running
1	Success
3	Failure
8	Paused
9	Pending

7.10.16.1 ?page=<n>&pageSize=<m>

You can limit the number of entries returned by a request by setting the page size and requesting a certain page. The response contains only the entries for that page, and includes links to URLs that can be used to retrieve other pages of information.

Request:

- Method: `GET`
- URL: `http://<baseURL>/publications?page=<n>&pageSize=<m>` .
  `page`: Query parameter used to get specific page.
  `pageSize`: Number of publication schedule instances per page.
- Body: none

Response: none

7.10.17 Getting Details of Publication and Modifying or Deleting Publication

You can get publication details in BI platform using the `/v1/publications/<publication_id>` URL.

- Use the `GET` method to get publication details
- Use the `PUT` method to modify the publication on BI platform
- Use the `DELETE` method to delete a publication

GET http://<baseURL>/v1/publications/<publication_id>

Make a `GET` request to `/v1/publications/<publication_id>` to retrive publication details.

Response:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <id>tag:sap.com,2010:bip-rs/infostore</id>
    <title type="text">Publication</title>
    <updated>2016-06-17T14:32:25.299Z</updated>
    <link href="http://localhost:6405/biprws/v1/publications?
page=1&amp;pagesize=50" rel="self"/>
    <link href="http://localhost:6405/biprws/v1/publications?
page=1&amp;pagesize=50" rel="first"/>
    <link href="http://localhost:6405/biprws/v1/publications?
page=1&amp;pagesize=50" rel="last"/>
    <entry>
        <title type="text">pub2</title>
        <id>tag:sap.com,2010:bip-rs/ARbvVAjpG8lIjxCSoUFXBRk</id>
        <author>
            <name>Administrator</name>
            <uri>http://localhost:6405/biprws/infostore/12</uri>
        </author>
        <updated>2016-05-19T14:22:24.590Z</updated>
        <link href="http://localhost:6405/biprws/v1/publications/7068"
rel="alternate"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">ARbvVAjpG8lIjxCSoUFXBRk</attr>
                <attr name="keywords" type="string"></attr>
                <attr name="name" type="string">pub2</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">7068</attr>
                <attr name="parentid" type="string">6102</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

PUT http://<baseURL>/v1/publications/<publication_id>.

To modify the details of a publication, use `PUT` request to `/publications/<publication_id>` URL.

Request:

- Method: `PUT`
- URL: `http://<baseURL>/v1/publications/<publication_id>`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Body: /publications/<publication_id>

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/infostore/12</uri>
    </author>
    <title type="text">pub2_cr</title>
    <updated>2016-06-17T14:33:34.611Z</updated>
    <link href="http://localhost:6405/biprws/v1/publications/6279"/>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">Aa.IfYf_S3RNmgwsgx0o5IA</attr>
            <attr name="enterpriseusergroups" type="int32">0</attr>
            <attr name="keywords" type="string"></attr>
            <attr name="documents" type="int32">0</attr>
            <attr name="destinations" type="int32">1</attr>
```

```
            <attr name="name" type="string">pub2_cr</attr>
            <attr name="description" type="string"></attr>
            <attr name="enterpriseusers" type="int32">0</attr>
            <attr name="id" type="string">6279</attr>
            <attr name="dynamicrecipients" type="bool">false</attr>
            <attr name="parentid" type="string">6102</attr>
        </attrs>
    </content>
</entry>
```

- Use `<attr name="enterpriseusergroups" type="int32">3</attr>` gives you the count of the number of enterprise user groups added to the publication.
- Use `<attr name="keywords" type="string">keyword</attr>` to define publication keyword.
- Use `<attr name="documents" type="string">Keyword</attr>` gives you the count of the number of documents added to the publication.
- Use `<attr name="destinations" type="string">[CrystalEnterprise.Smtp, CrystalEnterprise.Managed]</attr>` gives you the collection of enabled destinations for the publication.
- Use `<attr name="name" type="string">pub2_cr</attr>` to define publication name.
- Use `<attr name="description" type="string"></attr>` to enter description.
- Use `<attr name="enterpriseusers" type="int32">25</attr>` gives you the count of the number of enterprise users added to the publication.
- Use `<attr name="dynamicrecipients" type="string">true</attr>` gives you the boolean value whether the dynamic recipients are enabled or not.
- Use `<attr name="id" type="string">6279</attr>` to enter id.
- Use `<attr name="parentid" type="string">6102</antr>` to enter parent id.

- Response:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/infostore/12</uri>
    </author>
    <title type="text">pub2_cr</title>
    <updated>2016-06-17T14:45:40.556Z</updated>
    <link href="http://localhost:6405/biprws/v1/publications/6279"/>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">Aa.IfYf_S3RNmgwsgx0o5IA</attr>
            <attr name="keywords" type="string"></attr>
            <attr name="name" type="string">pub2_cr</attr>
            <attr name="description" type="string"></attr>
            <attr name="id" type="string">6279</attr>
            <attr name="parentid" type="string">6102</attr>
        </attrs>
    </content>
</entry>
```

DELETE method

Delete a publication, using `http://<baseURL>/v1/publications/<publication_id>` URL.

Request:

- Method: DELETE
- URL: `DELETE http://<baseURL>/v1/publications/<publication_id>`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Body:
  An XML feed displays success or failure of deletion of a publication. This example shows the deletion of a user for the resource with ID=6103.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Publication</title>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.10.18 Configuring Merged PDF Settings

To configure merged PDF settings in the publication, follow the methods as described below:

GET http://<host>:<port>/biprws/v1/publications/<publication_id>/mergedpdfoptions

Use the `GET` method using the following details, to fetch the template body:

- URL: `http://<host>:<port>/biprws/v1/publications/<publication_id>/mergedpdfoptions`

  Note

  `<publication_id>` is the ID of publication that you want to publish as a PDF.

- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: Nil
  RESULT: You receive a template body from the server.The following is an example of a response received from the server:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="tableOfContentsItemFont" type="string">TIMES_ROMAN</attr>
```

```
            <attr name="pageNumberFontSize" type="real">12.0</attr>
            <attr name="restrictionsUserPassword" type="string"></attr>
            <attr name="restrictionsAllowCopyPaste" type="bool">false</attr>
            <attr name="restrictionsOwnerPassword" type="string"></attr>
            <attr name="createRestrictions" type="bool">false</attr>
            <attr name="pageNumberFont" type="string">TIMES_ROMAN</attr>
            <attr name="tableOfContentsTitleFont" type="string">TIMES_ROMAN</
attr>
            <attr name="pageNumberFormat" type="string">Page &amp;p of &amp;P</
attr>
            <attr name="createTableOfContents" type="bool">true</attr>
            <attr name="createRunningPageNumbers" type="bool">false</attr>
            <attr name="tableOfContentsItemFontColor" type="string">BLACK</attr>
            <attr name="restrictionsAllowPrinting" type="bool">false</attr>
            <attr name="restrictionsAllowModificationOfContents"
type="bool">false</attr>
            <attr name="pageNumberAppyToTableOfContents" type="bool">false</attr>
            <attr name="restrictionsAllowAnnotationModification"
type="bool">false</attr>
            <attr name="pageNumberFontColor" type="string">BLACK</attr>
            <attr name="tableOfContentsTitleFontSize" type="real">16.5</attr>
            <attr name="tableOfContentsItemFontSize" type="real">12.0</attr>
            <attr name="tableOfContentsTitleFontColor" type="string">BLACK</attr>
            <attr name="pageNumberLocation" type="string">BOT_MIDDLE</attr>
            <attr name="tableOfContentsTitle" type="string"></attr>
        </attrs>
    </content>
</entry>
```

POST http://<host>:<port>/biprws/v1/publications/<publication_id>/mergedpdfoptions

Use the `POST` method to post the merged PDF options for your publication to the server with the following details:

- URL & header: Same as mentioned in the GET method.
- Body: Pass the required attribute values (such as `restrictionsAllowPrinting`, `restrictionsAllowModificationOfContents` boolean parameters) as shown in the below example:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="tableOfContentsItemFont" type="string">TIMES_ROMAN</attr>
            <attr name="pageNumberFontSize" type="real">12.0</attr>
            <attr name="restrictionsUserPassword" type="string"></attr>
            <attr name="restrictionsAllowCopyPaste" type="bool">false</attr>
            <attr name="restrictionsOwnerPassword" type="string"></attr>
            <attr name="createRestrictions" type="bool">false</attr>
            <attr name="pageNumberFont" type="string">TIMES_ROMAN</attr>
            <attr name="tableOfContentsTitleFont" type="string">TIMES_ROMAN</
attr>
            <attr name="pageNumberFormat" type="string">Page &amp;p of &amp;P</
attr>
            <attr name="createTableOfContents" type="bool">true</attr>
            <attr name="createRunningPageNumbers" type="bool">false</attr>
            <attr name="tableOfContentsItemFontColor" type="string">BLACK</attr>
            <attr name="restrictionsAllowPrinting" type="bool">true</attr>
            <attr name="restrictionsAllowModificationOfContents"
type="bool">true</attr>
            <attr name="pageNumberAppyToTableOfContents" type="bool">false</attr>
            <attr name="restrictionsAllowAnnotationModification"
type="bool">false</attr>
```

```
            <attr name="pageNumberFontColor" type="string">BLACK</attr>
            <attr name="tableOfContentsTitleFontSize" type="real">16.5</attr>
            <attr name="tableOfContentsItemFontSize" type="real">12.0</attr>
            <attr name="tableOfContentsTitleFontColor" type="string">BLACK</attr>
            <attr name="pageNumberLocation" type="string">BOT_MIDDLE</attr>
            <attr name="tableOfContentsTitle" type="string"></attr>
        </attrs>
    </content>
</entry>
```

- **Note**

  1. For attributes associated with `<Table of Contents>`, `<Page Numbers>` and `<Restrictions>` to be applicable in your publication, **set the below parameters to 'true'** in the body.

     Mandatory Attributes for Merged PDF

Attribute	Description
`createTableOfContents`	Creates a table of contents in the PDF
`createRunningPageNumbers`	Creates page numbers in the PDF
`createRestrictions`	Applies restrictions or security

  2. Valid options for:

     Font attributes- ARIAL, COURIER, HELVETICA, TIMES_ROMAN
     Color attributes- BLACK, BLUE
     Position attributes- TOP_MIDDLE, TOP_RIGHT, BOT_RIGHT, BOT_MIDDLE, BOT_LEFT, TOP_LEFT

RESULT: The merged PDF settings for your publication are posted to the server and you receive a message indicating success.

DELETE http://<host>:<port>/biprws/v1/publications/<publication_id>/mergedpdfoptions

To delete merged PDF settings for your publication, use the `DELETE` method with the same URL and header as described in the above methods. Nothing is required to be passed in the body.

Caution

Only SAP Crystal Report documents are supported for the PDF publishing. PDF merge options are **not supported** for publications with source documents **other than Crystal Reports**.

7.10.19 Configuring the Scheduling Server Group

To configure the scheduling server group for a publication, follow the methods as described below:

GET http://<baseURL>/v1/publications/<publication_id>/schedulingservergroup

- Use the `GET` method using the following details, to fetch the template body:
  - URL: `http://<baseURL>/v1/publications/<publication_id>/schedulingservergroup`

    > **Note**
    >
    > `<publication_id>` is the ID of publication.

  - Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

  - Body: Nil
    RESULT: You receive a template in response from the server. The following is an example of a response template:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="schedulingType" type="string">FIRST_AVAILABLE</attr>
            <attr name="runAtOriginSite" type="bool">false</attr>
            <attr name="serverGroupId" type="int32">0</attr>
        </attrs>
    </content>
</entry>
```

POST http://<baseURL>/v1/publications/<publication_id>/schedulingservergroup

- Use the `POST` method to post the merged PDF options for your publication to the server with the following details:
  - URL & Header: Same as in GET method.
  - Body: Pass the required attribute values as shown in the below example:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
```

```
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="schedulingType" type="string">PREFERRED</attr>
            <attr name="runAtOriginSite" type="bool">false</attr>
            <attr name="serverGroupId" type="int32">0</attr>
        </attrs>
    </content>
</entry>
```

> **Tip**
>
> Valid values for `schedulingType` are "FIRST_AVAILABLE", "PREFERRED" and "SPECIFIED":
> - "FIRST_AVAILABLE" is essentially “unset”, since this means run on any server that is available.
> - "PREFERRED" and "SPECIFIED" refer to the respective values configured on the CMC.

RESULT: The scheduling server group for your publication is posted to the server and you receive a message indicating success.

7.10.20 Configuring Notifications For Scheduled Publications

If your publication is scheduled, you can define notifications to receive success or failure messages when the instances of the publication run on the server.

GET `http://<baseURL>/v1/publications/<publication_id>/notification/onfailure`

- Use the `GET` method using the following details, to fetch the template body:
  - URL: `http://<baseURL>/v1/publications/<publication_id>/notification/onfailure`

    > **Note**
    >
    > Use `onsuccess` in the URL to **configure notification for successful running** of the instance.

  - Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

  - Body: Nil
    RESULT: A template is received from the server. The following is an example of the template:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="useJobServerDefaults" type="bool">false</attr>
            <attr name="CC" type="string"></attr>
            <attr name="subject" type="string">FAILED</attr>
```

```
            <attr name="from" type="string">S</attr>
            <attr name="to" type="string">A</attr>
            <attr name="message" type="string">FAILED</attr>
        </attrs>
    </content>
</entry>
```

POST `http://<baseURL>/v1/publications/<publication_id>/notification/onfailure`

- Use the `POST` method to configure events for your publication with the following details:
  - URL & header: Same as in the GET method.
  - Body: Pass the required attribute values as shown in the below example:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="useJobServerDefaults" type="bool">false</attr>
            <attr name="CC" type="string"></attr>
            <attr name="subject" type="string">FAILED</attr>
            <attr name="from" type="string">S</antr>
            <attr name="to" type="string">A</attr>
            <attr name="message" type="string">FAILED</attr>
        </attrs>
    </content>
</entry>
```

Note

If `useJobServerDefaults` is set to 'true', **all attribute values are ignored** and the job server settings configured on the CMC are used.

RESULT: The specified notification is configured for your publication and you receive a message indicating success in the RESTful Web service client.

Tip

To fetch available notifications for your scheduled publication at any point in time, use GET method with the URL `http://<baseURL>/publications/<publication_id>/notification` and same header as used in the above procedure. Below is an example of the response you receive from the server:

Sample Code

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <title type="text">onsuccess</title>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6258/
notification/onsuccess" rel="alternate"/>
    </entry>
    <entry>
        <title type="text">onfailure</title>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6258/
notification/onfailure" rel="alternate"/>
    </entry>
</feed>
```

DELETE `http://<baseURL>/v1/publications/<publication_id>/notification/onsuccess(onfailure)`

- Use the `DELETE` method to delete the existing events configured for your publication:
  - URL & header: Same as in GET/POST method.
  - Body: Nil

  RESULT: The notifications configured for your scheduled publication are deleted on the server, and you receive a message indicating success.

7.10.21 Configuring Advanced Settings

You can configure a few advanced settings for your publication using the methods which are described here.

GET `http://<baseURL>/v1/publications/<publication_id>/advanced`

- Use the `GET` method using the following details, to fetch the template body:
  - URL: `http://<baseURL>/v1/publications/<publication_id>/advanced`

    > **Note**
    >
    > `<publication_id>` is the ID of publication.

  - Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

  - Body: Nil

    RESULT: You receive a request template from the server. The following is an example of the template received from the server:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="mergeProfiles" type="bool">false</attr>
            <attr name="scopeBatchSize" type="int32">50</attr>
            <attr name="reportBurstingMethod" type="string">FETCH_PER_BAT</
attr>
        </attrs>
    </content>
</entry>
```

POST `http://<baseURL>/v1/publications/<publication_id>/advanced`

- Use the `POST` method to configure events for your publication with the following details:
  - URL & Header: Same as described in GET method.
  - Body: Pass the required attribute values as shown in the below example:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="mergeProfiles" type="bool">true</attr>
            <attr name="scopeBatchSize" type="int32">50</attr>
            <attr name="reportBurstingMethod" type="string">FETCH_PER_BAT</
attr>
        </attrs>
    </content>
</entry>
```

Note

Advanced settings closely follow the CMC and are described below:

Advanced Setting Attributes

Attribute	Valid Values	Meaning
`mergeprofiles`	`true, false`	Corresponds to the “Profile Resolution” header in the advanced settings in CMC: ○ If 'false' (default): Do not merge (distinct profiles from multiple parent user groups result in separate documents). ○ If 'true': Merge (distinct profiles from multiple parent user groups apply to the same document).
`scopebatchsize`	`default` (50), `custom` (can be any numeric value less than or equal to 50) **Note** `scopebatchsize` cannot be larger than 50.	Corresponds to the "Scope Batch Size Limit" in the CMC.

Attribute	Valid Values	Meaning
reportBurstingMethod	FETCH_FOR_ALL, FETCH_PER_BAT, FETCH_PER_REC	Corresponds to the "Report Bursting Method" in the CMC inferring the frequency of database fetch: ○ FETCH_FOR_ALL: One database fetch for all recipients (recommended for minimizing the number of database queries) ○ FETCH_PER_BAT: One database fetch for each batch of recipients (recommended for high-volume scenarios) ○ FETCH_PER_REC: One database fetch per recipient (recommended when using row-level security in Universes or Business Views)

RESULT: The advanced settings are configured for your publication and you receive a message indicating success in the RESTful Web client page.

7.10.22 Configuring Events For Scheduled Publications

You can associate events with a scheduled publication. This means that you can define if you want a publication instance to wait for a specific event before running or not. To configure events for your scheduled publications, follow the methods described in this topic:

GET `http://<baseURL>/v1/publications/<publication_id>/events`

- Use the `GET` method using the following details, to fetch the template body:
  - URL: `http://<baseURL>/v1/publications/<publication_id>/events`

    > **Note**
    >
    > `<publication_id>` is the ID of publication that you want to publish as a PDF.

  - Header:

Name	Value
Content-Type	application/xml

Name	Value
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: Nil
  RESULT: You receive the template body. The following is an example template:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="eventId" type="int32">0</attr>
                <attr name="waitForEvent" type="bool">true</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

POST `http://<baseURL>/v1/publications/<publication_id>/events`

- Use the `POST` method to configure events for your publication with the following details:
  - URL & header: Same as in GET method.
  - Body: Pass the required attribute values as shown in the below example:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="eventId" type="int32">8001</attr>
                <attr name="waitForEvent" type="bool">true</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

RESULT: The specified event is configured for your publication and you receive a message indicating success.

Tip

- `eventID` is the ID of your specific event.
- The `waitForEvent` property controls whether the event is going to be triggered (false) after the publication instances run, or the publication instances wait for the event (true).
- **Only system events** can be triggered.
- To delete the event settings for your publication, use the `DELETE` method with the header, URL, and body as specified in the procedure.

DELETE `http://<baseURL>/v1/publications/<publication_id>/events`

- Use the `DELETE` method to delete the existing events configured for your publication:
  - URL & header: Same as in GET method.
  - Body: Nil

  RESULT: The events configured for your scheduled publication are deleted on the server, and you receive a message indicating success.

7.10.23 Configuring Delivery Rules and Conditions

You can configure your publication such that it is sent to the recipients on the basis of conditions or rules. The RESTful APIs to configure delivery rules and conditions for a publication are described in the sub-chapters.

To fetch the currently available rules and conditions over a publication, follow the procedure described below:

GET `http://<baseURL>/v1/publications/<PUB ID>/recipientdelivery`

Use the `GET` method with the URL `http://<baseURL>/v1/publications/<PUB ID>/recipientdelivery` and the following header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

Below is an example of the response you receive from the server (**displays both- the available rule and condition**):

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://10.160.199.73:6405/biprws/infostore/12</uri>
    </author>
    <id>6258</id>
    <updated>2017-03-17T05:50:38.012Z</updated>
    <entry>
        <title type="text">condition</title>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6258/
recipientdelivery/condition" rel="alternate"/>
    </entry>
    <entry>
        <title type="text">rule</title>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6258/
recipientdelivery/rule" rel="alternate"/>
    </entry>
</feed>
```

7.10.23.1 Configuring Global Delivery Rules

You can configure global delivery rules **for publications with Crystal Reports for Enterprise or Crystal Reports 20xx source documents.**

GET `http://<baseURL>/v1/publications/<publication_id>/globaldeliveryrule`

- Use the `GET` method using the following details, to fetch the template body:
  - URL: `http://<baseURL>/v1/publications/<publication_id>/globaldeliveryrule`

    > **Note**
    >
    > `<publication_id>` is the ID of publication that you want to publish.

  - Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

  - Body: Nil
    You receive a template body as shown below.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="reportAlertDeliveryRule" type="string"></attr>
                <attr name="reportId" type="int32">7499</attr>
                <attr name="alertTestCondition" type="bool">false</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

- Copy the template response for use in the POST method.

POST `http://<baseURL>/v1/publications/<publication_id>/globaldeliveryrule`

- Use the `POST` method to configure events for your publication with the following details:
  - URL & header: Same as earlier.
  - Body: Paste the body received in the GET method and pass the required attribute values:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
```

```
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="reportAlertDeliveryRule" type="string"></attr>
                <attr name="reportId" type="int32">7499</attr>
                <attr name="alertTestCondition" type="bool">true</attr>

            </attrs>
        </content>
    </entry>
</feed>
```

Note

Delivery Rule Attributes

Attribute	Valid Values	Meaning
`reportId`	ID (no restriction)	Corresponds to the ID of source document in the publication.
`constantDeliveryRule`	`ALWAYS, NEVER, ONLY_IF_DATA, USE_RPT_RULE, REFRESH`	Defines the delivery rule. Each value corresponds to a CMC option. **Caution** ○ `REFRESH` applies only to Web Intelligence documents. ○ `USE_RPT_RULE` only applies to Crystal Reports 20xx documents. ○ With `USE_RPT_RULE`, you need to specify the next 2 parameters described below (`reportAlertDeliveryRule` and `alertTestCondition`). If their values aren't specified, the server **ignores** `constantDeliveryRule` altogether.
`reportAlertDeliveryRule`	Report alert (no value restriction)	Specifies the report alert for a Crystal Reports 20xx document. Only applicable when `constantDeliveryRule="USE_RPT_RULE"`
`alertTestCondition`	`true, false`	Specifies if the alert result should be 'true' or 'false'. Only applicable when `constantDeliveryRule="USE_RPT_RULE"`

RESULT: The global delivery rules are configured for your publication and you receive a message indicating success in the RESTful Web client page.

DELETE `http://<baseURL>/v1/publications/<publication_id>/globaldeliveryrule`

- Use the `DELETE` method using the following details, to fetch the template body:
  - URL: `http://<baseURL>/v1/publications/<publication_id>/globaldeliveryrule`

    > **Note**
    >
    > `<publication_id>` is the ID of publication that you want to publish.

  - Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

  - Body: Nil

RESULT: The existing global delivery rules are deleted for your publication and you receive a message indicating success in the RESTful Web client page.

> **Caution**
>
> Source documents other than Crystal reports are not supported for the global delivery rules' API.

7.10.23.2 Configuring Recipient Delivery Rules

7.10.23.2.1 Rules

GET `http://<baseURL>/v1/publications/<publication_id>/recipientdelivery/rule`

- Use the `GET` method using the following details, to fetch the template body:
  - URL: `http://<baseURL>/v1/publications/<publication_id>/recipientdelivery/rule`

    > **Note**
    >
    > `<publication_id>` is the ID of publication that you want to publish.

- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: Nil
  You receive a template body as shown below.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="reportAlertDeliveryRule" type="string"></attr>
                <attr name="reportId" type="int32">7499</attr>
                <attr name="alertTestCondition" type="bool">false</attr>
                <attr name="constantDeliveryRule" type="string">ALWAYS</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

- Copy the template response for use in the POST method.

POST `http://<baseURL>/v1/publications/<publication_id>/recipientdelivery/rule`

- Use the `POST` method to configure events for your publication with the following details:
  - URL & header: Same as earlier.
  - Body: Paste the body received in the GET method and pass the required attribute values:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="reportAlertDeliveryRule" type="string"></attr>
                <attr name="reportId" type="int32">7499</attr>
                <attr name="alertTestCondition" type="bool">true</attr>
                <attr name="constantDeliveryRule" type="string">ALWAYS</antr>
            </attrs>
        </content>
    </entry>
</feed>
```

Note

Delivery Rule Attributes

Attribute	Valid Values	Meaning
`reportId`	ID (no restriction)	Corresponds to the ID of source document in the publication.

Attribute	Valid Values	Meaning
`constantDeliveryRule`	`ALWAYS, NEVER, ONLY_IF_DATA, USE_RPT_RULE, REFRESH`	Defines the delivery rule. Each value corresponds to a CMC option. ⚠ Caution ○ `REFRESH` applies only to Web Intelligence documents. ○ `USE_RPT_RULE` only applies to Crystal Reports 20xx documents. ○ With `USE_RPT_RULE`, you need to specify the next 2 parameters described below (`reportAlertDeliveryRule` and `alertTestCondition`). If their values aren't specified, the server **ignores** `constantDeliveryRule` altogether.
`reportAlertDeliveryRule`	Report alert (no value restriction)	Specifies the report alert for a Crystal Reports 20xx document. Only applicable when `constantDeliveryRule="USE_RPT_RULE"`
`alertTestCondition`	`true, false`	Specifies if the alert result should be 'true' or 'false'. Only applicable when `constantDeliveryRule="USE_RPT_RULE"`

RESULT: The delivery rules are configured for your publication and you receive a message indicating success in the RESTful Web client page.

7.10.23.2.2 Conditions

GET `http://<baseURL>/publications/<publication_id>/recipientdelivery/condition`

- Use the `GET` method using the following details, to fetch the template body:

- URL: `http://<baseURL>/publications/<publication_id>/recipientdelivery/condition`

  > **Note**
  >
  > `<publication_id>` is the ID of publication that you want to publish.

- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: Nil
  RESULT: You receive a template response as shown below:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="deliverIfIndividualConditionsMet" type="bool">false</
attr>
        </attrs>
    </content>
</entry>
```

POST `http://<baseURL>/publications/<publication_id>/recipientdelivery/condition`

- Use the `POST` method with the following details:
  - URL & header: Same as GET method.
  - Body: Pass the required attribute values:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="deliverIfIndividualConditionsMet" type="bool">true</
attr>
        </attrs>
    </content>
</entry>
```

> **Tip**
>
> The `deliverIfIndividualConditionsMet` property corresponds to the *Recipient Delivery Rule* radio-button field in the CMC as follows:
>
> - if 'true': Deliver individual document when condition is met
> - if 'false': Deliver all documents only when all conditions are met

RESULT: The delivery condition is configured for your publication and you receive a message indicating success in the RESTful Web client page.

7.10.24 Fetching Distribution Failure Information

You can fetch distribution failure information for your scheduled publications on the global level (including all recipients), and for individual recipients (dynamic/enterprise) as described in the sub-chapters.

GET `http://<baseURL>/v1/publications/<publication_id>/distributionfailures`

Use the `GET` method with the URL `http://<baseURL>/v1/publications/<publication_id>/distributionfailures` and the following header:

> **Note**
>
> `<publication_id>` is the ID of publication that you want to publish.

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

Below is an example of the response you receive from the server (**displays both- the available rule and condition**):

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://10.160.199.73:6405/biprws/infostore/12</uri>
    </author>
    <id>6258</id>
    <updated>2017-03-17T06:12:54.689Z</updated>
    <entry>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6258/
distributionfailures/dynamic" rel="alternate"/>
    </entry>
    <entry>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6258/
distributionfailures/enterprise" rel="alternate"/>
    </entry>
</feed>
```

7.10.24.1 Fetching Dynamic Recipient Failures

GET `http://<baseURL>/v1/publications/<PUB ID>/distributionfailures/dynamic`

Follow the below procedure:

1. Use the `GET` method using the following details, to fetch the template body:
   - URL: `http://<baseURL>/v1/publications/<publication_id>/distributionfailures/dynamic`

     > **Note**
     >
     > `<publication_id>` is the ID of publication that you want to publish.

   - Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

   - Body: Nil
     The following is an example of a response template received from the server if you fetch dynamic failures using the URL `http://<baseURL>/v1/publications/<publication_id>/distributionfailures/dynamic`:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://10.160.199.73:6405/biprws/infostore/12</uri>
    </author>
    <id>6258</id>
    <updated>2017-03-17T06:12:54.689Z</updated>
    <entry>
        <title type="text">fpub</title>
        <updated>2017-02-07T19:35:56.581Z</updated>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6258/
distributionfailures/dynamic/instance/6511" rel="alternate"/>
    </entry>
    <entry>
        <title type="text">fpub</title>
        <updated>2017-02-07T20:06:17.049Z</updated>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6258/
distributionfailures/dynamic/instance/6553" rel="alternate"/>
    </entry>
    <entry>
        <title type="text">fpub</title>
        <updated>2017-02-07T20:39:33.535Z</updated>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6258/
distributionfailures/dynamic/instance/6625" rel="alternate"/>
    </entry>
    <entry>
        <title type="text">fpub</title>
        <updated>2017-02-08T21:17:58.095Z</updated>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6258/
distributionfailures/dynamic/instance/7847" rel="alternate"/>
    </entry>
    <entry>
        <title type="text">fpub</title>
        <updated>2017-02-08T22:20:34.768Z</updated>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6258/
distributionfailures/dynamic/instance/7973" rel="alternate"/>
    </entry>
    <entry>
        <title type="text">fpub</title>
```

```
        <updated>2017-02-08T22:52:22.268Z</updated>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6258/
distributionfailures/dynamic/instance/8007" rel="alternate"/>
    </entry>
    <entry>
        <title type="text">fpub</title>
        <updated>2017-02-09T12:08:36.094Z</updated>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6258/
distributionfailures/dynamic/instance/8879" rel="alternate"/>
    </entry>
    <entry>
        <title type="text">fpub</title>
        <updated>2017-02-09T12:12:30.126Z</updated>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6258/
distributionfailures/dynamic/instance/8920" rel="alternate"/>
    </entry>
    <entry>
        <title type="text">fpub</title>
        <updated>2017-02-09T12:38:53.638Z</updated>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6258/
distributionfailures/dynamic/instance/9180" rel="alternate"/>
    </entry>
    <entry>
        <title type="text">fpub</title>
        <updated>2017-02-09T12:43:08.168Z</updated>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6258/
distributionfailures/dynamic/instance/9208" rel="alternate"/>
    </entry>
</feed>
```

7.10.24.2 Fetching Enterprise Recipient Failures

GET `http://<baseURL>/v1/publications/<PUB ID>/distributionfailures/enterprise`

Follow the below procedure:

1. Use the `GET` method using the following details, to fetch the template body:
   - URL: `http://<baseURL>/v1/publications/<publication_id>/distributionfailures/enterprise`

     Note

     `<publication_id>` is the ID of publication that you want to publish.

   - Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: Nil
  The following is an example of a response template received from the server if you fetch dynamic failures using the URL `http://<baseURL>/v1/publications/<publication_id>/distributionfailures/enterprise`:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://10.160.199.73:6405/biprws/infostore/12</uri>
    </author>
    <id>6258</id>
    <updated>2017-03-17T06:12:54.689Z</updated>
    <entry>
        <title type="text">fpub</title>
        <updated>2017-02-07T19:35:56.581Z</updated>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6987/
distributionfailures/enterprise/instance/6511" rel="alternate"/>
    </entry>
    <entry>
        <title type="text">fpub</title>
        <updated>2017-02-07T20:06:17.049Z</updated>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6987/
distributionfailures/enterprise/instance/6553" rel="alternate"/>
    </entry>
    <entry>
        <title type="text">fpub</title>
        <updated>2017-02-07T20:39:33.535Z</updated>
        <link href="http://10.160.199.73:6405/biprws/v1/publications/6987/
distributionfailures/enterprise/instance/6625" rel="alternate"/>
    </entry>
</feed>
```

7.10.25 Personalizing Publications

You can personalize publications on the BI platform using the `v1/publications/id/personalization` URL.

GET http://<baseURL>/v1/publications/id/personalization

Make a `GET` request to `v1/publications/id/publication` URL to personalize publication for global and local profiles.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/publications/id/personalization`
- Body: none

Response:

You can see an XML feed of personalization for global and local profiles.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <author>
```

```
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/infostore/12</uri>
    </author>
    <id>6493</id>
    <updated>2016-03-17T14:04:23.267Z</updated>
    <entry>
        <title type="text">Global Profiles</title>
        <link href="http://localhost:6405/biprws/v1/publications/6493/
personalization/globalProfiles" rel="alternate" />
    </entry>
    <entry>
        <title type="text">Local Profiles</title>
        <link href="http://localhost:6405/biprws/v1/publications/6493/
personalization/localProfiles" rel="alternate" />
    </entry>
</feed>
```

7.10.25.1 Global Profiles

You can personalize publication for global profiles on the BI platform.

GET http://<baseURL>/v1/publications/id/personalization/globalprofiles

Make a `GET` request to `v1/publications/id/personalization/globalprofiles` URL to personalize publication for enterprise recipients.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/publications/id/personalization/globalprofiles`
- Body: none

Response:

You can see an XML feed of personalization option for enterprise recipients under global profiles.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/infostore/12</uri>
    </author>
    <id>6493</id>
    <updated>2016-03-17T14:04:23.267Z</updated>
    <entry>
        <title type="text">Enterprise Recipient</title>
        <link href="http://localhost:6405/biprws/v1/publications/6493/
personalization/globalprofiles/enterpriserecipient" rel="alternate" />
    </entry>
</feed>
```

7.10.25.1.1 Enterprise Recipients - Global Profiles

You can personalize publication for enterprise recipients on BI platform.

GET http://<baseURL>/v1/publications/<publication_id>/personalization/globalprofiles/enterpriserecipients

Make a `GET` request to `v1/publications/<publication_id>/personalization/globalprofiles/enterpriserecipients` URL to list the personalized publication for enterprise recipients.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/publications/id/personalization/globalprofiles/enterpriserecipients`
- Body: none

Response:

You can see an XML feed of publication personalized for enterprise recipients.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="reportID" type="int32">5348</attr>
                <attr name="profileID" type="int32">7319</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="reportID" type="int32">5346</attr>
                <attr name="profileID" type="int32">7319</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="reportID" type="int32">5350</attr>
                <attr name="profileID" type="int32">7320</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

POST http://<baseURL>/v1/publications/<publication_id>/personalization/globalprofiles/enterpriserecipients

Make a `POST` request to `v1/publications/<publication_id>/personalization/globalprofiles/enterpriserecipients` URL to send the publication for enterprise recipients.

Request:

- Method: POST
- URL: `http://<baseURL>/v1/publications/<publication_id>/personalization/globalprofiles/enterpriserecipients`
- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="reportID" type="int32">report_Id</attr>
                <attr name="profileID" type="int32">profile_id</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

  - Use `<attr name="reportID" type="int32">report_Id</attr>` to define the Id of the report.
  - Use `<attr name="profileID" type="int32">profile_id</antr>` to define the Id of the profile.

Response:

You can see an XML feed of publication personalized for enterprise recipients.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="reportID" type="int32">5350</attr>
                <attr name="profileID" type="int32">7319</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

DELETE http://<baseURL>/v1/publications/<publication_id>/personalization/globalprofiles/enterpriserecipients

Make a `DELETE` request to `v1/publications/id/personalization/globalprofiles/enterpriserecipients` URL to delete the personalized publication for enterprise recipients.

Request:

- Method: DELETE
- URL: `http://<baseURL>/v1/publications/<publication_id>/personalization/globalprofiles/enterpriserecipients`

- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="reportID" type="int32">report_Id</attr>
                <attr name="profileID" type="int32">profile_id</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

  - Use `<attr name="reportID" type="int32">report_Id</attr>` to define the Id of the report.
  - Use `<attr name="profileID" type="int32">profile_id</antr>` to define the Id of the profile.

Response:

An XML feed displays the success or failure message for delete operation. This example shows the successful deletion of the personalized publication for a particular report Id and profile Id

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="reportID" type="int32">5350</attr>
                <attr name="profileID" type="int32">7319</attr>
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.10.25.2 Local Profiles

You can personalize publication for local profiles on the BI platform.

GET http://<baseURL>/v1/publications/id/personalization/localprofiles

Make a `GET` request to `v1/publications/id/personalization/localprofiles` URL to personalize publication for enterprise recipients.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/publications/id/personalization/localprofiles`
- Body: none

Response:

You can see an XML feed of personalization options for enterprise recipients under local profiles.

```
<feed xmlns="http://www.w3.org/2005/Atom">
```

```
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/infostore/12</uri>
    </author>
    <id>7297</id>
    <updated>2016-03-21T05:24:48.015Z</updated>
    <entry>
        <title type="text">Enterprise Recipient</title>
        <link href="http://localhost:6405/biprws/v1/publications/6493/
personalization/localprofiles/enterpriserecipient" rel="alternate" />
    </entry>
    <entry>
        <title type="text">Dynamic Recipient</title>
        <link href="http://localhost:6405/biprws/v1/publications/6493/
personalization/localprofiles/dynamicrecipient" rel="alternate" />
    </entry>
</feed>
```

7.10.25.2.1 Dynamic Recipients - Local Profiles

You can personalize publication for recipients on BI platform.

GET `http://<baseURL>/v1/publications/<publication_id>/personalization/localprofiles/dynamicrecipients`

Make a `GET` request to `v1/publications/<publication_id>/personalization/localprofiles/dynamicrecipients` URL to list the personalized publications for dynamic recipients.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/publications/<publication_id>/personalization/localprofiles/dynamicrecipients`
- Body: none

Response:

You can see an XML feed of publication personalized for dynamic recipients.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="mapping" type="string">Sheet1_.Country</attr>
                <attr name="subreportName" type="string"></attr>
                <attr name="sourceReportId" type="int32">7499</attr>
                <attr name="reportField" type="string">{Sheet1_.Country}</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="mapping" type="string">Sheet1_.Country</attr>
```

```
                <attr name="subreportName" type="string">s1</attr>
                <attr name="sourceReportId" type="int32">7499</attr>
                <attr name="reportField" type="string">{?subcity}</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

> **Note**
>
> **Changes to the Personalization Feed Template Attributes in 4.2 SP4**
>
> - All attributes that are not required to set the properties have been removed from the response (e.g Report Name & Report CUID).
> - A new property `<subreportName>` has been added which allows prompts in subreports to be configured. [To configure prompts or fields in the main report, just leave this as an empty string.]
> - The `<mapping>` property corresponds to the column in the `<Dynamic Recipient Provider>` document.
> - The `<reportField>` property corresponds to the field or prompt that will receive the data from the Data Provider document.
> - The `<sourceReportId>` is the source document to which the mapping corresponds to within the publication.

POST `http://<baseURL>/v1/publications/<publication_id>/personalization/localprofiles/dynamicrecipients`

Request:

- Method: POST
- URL: `http://<baseURL>/v1/publications/<publication_id>/personalization/localprofiles/dynamicrecipients`
- Body: Pass the required attribute values as shown in the example below:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="mapping" type="string">Sheet2_.Country</attr>
                <attr name="subreportName" type="string"></attr>
                <attr name="sourceReportId" type="int32">7499</attr>
                <attr name="reportField" type="string">{Sheet2_.Country}</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="mapping" type="string">Sheet1_.Country</attr>
                <attr name="subreportName" type="string">s2</attr>
                <attr name="sourceReportId" type="int32">7499</attr>
                <attr name="reportField" type="string">{?region}</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

Response:

You can see an XML feed of publications personalized for dynamic recipients.

DELETE http://<baseURL>/v1/publications/<publication_id>/personalization/localprofiles/dynamicrecipients

Make a `DELETE` request to `v1/publications/<publication_id>/personalization/localprofiles/dynamicrecipients` URL to delete the personalized publication for dynamic recipients.

Request:

- Method: DELETE
- URL: `http://<baseURL>/v1/publications/<publication_id>/personalization/localprofiles/dynamicrecipients`
- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="mapping" type="string">Sheet1_.Country</attr>
                <attr name="subreportName" type="string"></attr>
                <attr name="sourceReportId" type="int32">7499</attr>
                <attr name="reportField" type="string">{Sheet1_.Country}</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

Response:

An XML feed displays the success or failure message for delete operation. This example shows the successful deletion of the personalized publication for a particular `<sourcereportId>` and `<mapping>`.

7.10.25.2.2 Enterprise Recipients - Local Profiles

You can personalize publications for enterprise recipients on the BI platform.

GET `http://<baseURL>/v1/publications/<publication_id>/personalization/localprofiles/enterpriserecipients`

Make a `GET` request to `v1/publications/<publication_id>/personalization/localprofiles/enterpriserecipients` URL to list the personalized publication for enterprise recipients.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/publications/id/personalization/localprofiles/enterpriserecipients`
- Body: none

Response:

You can see an XML feed of publication personalized for enterprise recipients.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="subreportName" type="string"></attr>
                <attr name="sourceReportId" type="int32">6498</attr>
                <attr name="targetProfileId" type="string">6449</attr>
                <attr name="reportField" type="string">{Sheet1_.Country}</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

> **Note**
>
> **Changes to the Personalization Feed Template Attributes in 4.2 SP4**
>
> - All attributes which are not required to set the properties have been removed from the response (e.g Report Name & Report CUID).
> - A new property `<subreportName>` has been added which allows prompts in subreports to be configured. [To configure prompts or fields in the main report, just leave this as an empty string.]
> - The `<mapping>` property corresponds to the column in the Enterprise Recipient Provider document.
> - The `<targetProfileId>` property corresponds to the ID of the Enterprise Profile used in the mapping. [This attribute replaces the `<reportField>` property used for dynamic recipients.]
> - The `<sourceReportId>` is the source document to which the mapping corresponds to within the publication.

POST `http://<baseURL>/v1/publications/id/personalization/localprofiles/enterpriserecipients`

Make a `POST` request to `v1/publications/id/personalization/localprofiles/enterpriserecipients` URL to send the publication for enterprise recipients.

Request:

- Method: POST
- URL: `http://<baseURL>/v1/publications/id/personalization/localprofiles/enterpriserecipients`
- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
```

```
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="subreportName" type="string"></attr>
                <attr name="sourceReportId" type="int32">6498</attr>
                <attr name="targetProfileId" type="string">8976</attr>
                <attr name="reportField" type="string">{Sheet2_.Country}</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

Response:

You can see an XML feed of publication personalized for enterprise recipients.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="sourceReportId" type="int32">6498</attr>
                <attr name="targetProfileId" type="string">8976</attr>
                <attr name="reportField" type="string">{Sheet2_.Country}</attr>
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

DELETE http://<baseURL>/v1/publications/id/personalization/localprofiles/enterpriserecipients

Make a `DELETE` request to `v1/publications/id/personalization/localprofiles/enterpriserecipients` URL to delete the personalized publication for enterprise recipients.

Request:

- Method: DELETE
- URL: `http://<baseURL>/v1/publications/id/personalization/localprofiles/enterpriserecipients`
- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="sourceReportId" type="int32">6498</attr>
                <attr name="targetProfileId" type="string">8976</attr>
                <attr name="reportField" type="string">{Sheet1_.Country}</attr>
                </attrs>
        </content>
    </entry>
</feed>
```

Response:

An XML feed displays the success or failure message for delete operation. This example shows the successful deletion of the personalized publication for a particular report Id and profile Id

```
<feed xmlns="http://www.w3.org/2005/Atom">
```

```
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="sourceReportId" type="int32">6498</attr>
                <attr name="targetProfileId" type="string">8976</attr>
                <attr name="reportField" type="string">{Sheet1_.Country}</attr>
                <attr name="status" type="string">success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.11 Uploading and Downloading File

7.11.1 /infostore/folder/<folder_id>/file

http://<baseURL>/infostore/folder/<folder_id>/file

Make a `POST` request using, `/infostore/folder/<folder_id>/file` to upload a file.

Request:

1. Method: `POST`
2. URL: `http://<baseURL>/infostore/folder/<folder_id>/file`
   Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<folder_id>` with the SI_ID of the folder in which you want to upload a file.
3. Header:

Name	Value
`X-SAP-LogonToken`	The logon token value, in quotation marks.

4. Select *form-data*.

   > **Note**
   >
   > The workflow is executed in Postman client

5. Enter any keyword and select *Choose Files*.

   > **Note**
   >
   > The workflow is executed in Postman REST client

> **Note**
>
> - You can upload any BOE or Non-BOE documents.

- BOE files: Webi, Crystal and Lumira documents. These documents are uploaded as *Web Intelligence*, *Crystal Reports* and *Lumira Documents* respectively. Apart from these BOE documents, other documents are uploaded as *Agnostic* .
- Non-BOE files like: word, pdf, txt etc. These documents are uploaded as *Word Document*, *Adobe Acrobat* and *Text* respectively.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
Date	The date and time of the response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- **Body:**

```
<feed xmlns="http://www.w3.org/2005/Atom">
        <updated>2015-04-22T09:47:22.546Z</updated>
           <entry>
            <title type="text">test</title>
            <content type="application/xml">
                <attrs xmlns="http://www.sap.com/rws/bip">
                   <attr name="id" type="int32">21158</attr>
                   <attr name="title" type="string">test</attr>
                   <attr name="cuid" type="string">AfmTwykzQ71JgreADdNbDAw</attr>
                   <attr name="parentID" type="int32">5213</attr>
                 </attrs>
              </content>
          </entry>
          </feed>
```

7.11.2 /infostore/folder/<SI_ID>/file

GET http://<baseURL>/infostore/folder/<SI_ID>/file

Make a `GET` request to `/infostore/folder/<SI_ID>/file` to download a file.

> **Note**
>
> You can download these three BOE douments: Webi, Crystal and Lumira documents.

Request:

1. Method: `GET`
2. URL: `http://<baseURL>/infostore/folder/<SI_ID>/file`
   Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<SI_ID>` with the SI_ID of the file you want to download.

3. Header:

Name	Value
X-SAP-LogonToken	The logon token value, in quotation marks.

4. Body: none

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.
Content-Disposition	Type as attachment along with the filename

- Sample Code

  To download any files, you can use the following code snippet or you can customize the same as per your requirement.

```
         HttpURLConnection conn = (HttpURLConnection) url.openConnection();
         conn.setDoOutput(true);
         conn.setRequestMethod("GET"); //Request method (GET)
        conn.setRequestProperty("X-SAP-LOGONTOKEN" , "//LogonToken"); //Logon
token is the logon token for this session

         if (conn.getResponseCode() == 200)
         {
             InputStream inputStream = conn.getInputStream();
             OutputStream output = new FileOutputStream("//path to
download");  //Give a location to download the file
             byte[] buffer = new byte[1024];
             int bytesRead;
             while ((bytesRead = inputStream.read(buffer)) != -1)
               {
                output.write(buffer, 0, bytesRead);
               }
             output.close();
          }
          else
          {
           //Error
          }
```

Related Information

/infostore/folder/<folder_id>/file [page 259]

7.12 Managing Categories

This section provides you information on RESTful APIs to manage categories. Using these APIs, you can perform the following:

- Listing categories
- Creating category
- Getting details of the category
- Modifying details of the category
- Listing first level children under a category
- Listing first level document in a category
- Deleting category

7.12.1 Listing categories

GET http://<baseURL>/v1/categories

Get the list of categories using the `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/categories`
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Body:
  An XML feed of all the categories created in the BOE system is displayed.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <id>tag:sap.com,2010:bip-rs/infostore</id>
    <title type="text">CATEGORIES</title>
    <updated>2016-05-17T08:53:48.986Z</updated>
    <link href="http://10.160.205.83:6405/biprws/v1/categories?
page=1&amp;pagesize=50" rel="self"/>
    <link href="http://10.160.205.83:6405/biprws/v1/categories?
page=1&amp;pagesize=50" rel="first"/>
```

```
    <link href="http://10.160.205.83:6405/biprws/v1/categories?
page=1&amp;pagesize=50" rel="last"/>
    <entry>
        <title type="text">Administrator</title>
        <id>tag:sap.com,2010:bip-rs/CQEvAAAA9FZpPn7XfU2Ust.ltEwspg</id>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">CQEvAAAA9FZpPn7XfU2Ust.ltEwspg</
attr>
                <attr name="parentcuid" type="string">ATI2BcB9RGBFuBi5s1TwL7k</
attr>
                <attr name="name" type="string">Administrator</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">889</attr>
                <attr name="ownerid" type="string">12</attr>
                <attr name="type" type="string">PersonalCategory</attr>
                <attr name="updated" type="string">Tue May 17 00:01:54 PDT 2016</
attr>
                <attr name="parentid" type="string">47</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">Guest</title>
       <id>tag:sap.com,2010:bip-rs/CQEvAAAAyA4UZ_ELMkSD9WOc9qJ2Yg</id>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">CQEvAAAAyA4UZ_ELMkSD9WOc9qJ2Yg</
attr>
                <attr name="parentcuid" type="string">ATI2BcB9RGBFuBi5s1TwL7k</
attr>
                <attr name="name" type="string">Guest</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">883</attr>
                <attr name="ownerid" type="string">11</attr>
                <attr name="type" type="string">PersonalCategory</attr>
                <attr name="updated" type="string">Tue May 17 00:01:54 PDT 2016</
attr>
                <attr name="parentid" type="string">47</attr>
            </attrs>
        </content>
    </entry>
</feed>
```


Sorting categories

Sorting is the process of arranging the categories in a systematic order.

You can now sort categories sequentially in ascending or descending order based on the following attributes:

- Name
- Description
- ID of the category
- Type

Filtering categories

You can filter categories based on the *type* and *updated* attributes.

The query parameters for sort and filter are as follows:

Query Parameter

Parameter Name	Parameter Value
Sort	`sort=<+/-><attribute>` **Note** • For ascending order the value is + • For descending order the value is - • By default the categories are sorted in ascending based on name of the categories **Example** sort=+name
Filter	`type=<category>` • Type can take only two values either personal category or corporate category **Example** type=personal
Filter	`updated=<Start date year-month-dateThh:mm:ss.Timezone> , <End date year-month-dateThh:mm:ss.Timezone>` **Note** If end date is not specified, by default current date will be considered. **Example** updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z
Paging	• `page=<page number>` • `pagesize=<number>` `pagesize` is the number of objects displayed in response **Note** By default `page=1` and `pagesize=50` that is, in one page number 1, the top 50 objects will be displayed in response.

URL: `GET http://<baseURL>/v1/categories?sort=<name>&updated=<Start year-month-dateThh:mm:ss.727Z> , <End year-month-dateThh:mm:ss.727Z>&page=<page number>&pagesize=<number>`

Example

URL: `GET http://<baseURL>/v1/categories?sort=+name&updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z&page=2&pagesize=5`

once you send the request, the response displays the second page which has five objects and the list of categories updated between the specified date and sorted in ascending order.

7.12.2 Creating Categories

Create a category in the BI platform by making a request to the `/v1/categories/category` URL.

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to create a category in the BI platform.

GET http://<baseURL>/v1/categories/category

Make a `GET` request to `/v1/categories/category` to receive a template that should be used in the request body of the `POST` method to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/categories/category`
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="keywords" type="string"></attr>
            <attr name="name" type="string"></attr>
            <attr name="description" type="string"></attr>
            <attr name="type" type="string"></attr>
            <attr name="parentid" type="string"></attr>
        </attrs>
    </content>
</entry>
```

POST http://<baseURL>/v1/categories/category

To create a category, use the `POST` request to `/v1/categories/category` URL.

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/categories/categories`
- Header:

Name	Value
`Content Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="keywords" type="string"></attr>
            <attr name="name" type="string">demo1</attr>
            <attr name="description" type="string"></attr>
            <attr name="type" type="string">Personal</attr>
            <attr name="parentid" type="string">889</antr>
        </attrs>
    </content>
</entry>
```

  - Use `<attr name="keywords" type="string">keywords</attr>` to define keywords.
  - Use `<attr name="name" type="string">name</attr>` to define the name.
  - Use `<attr name="description" type="string">dscription</attr>` to define a description.
  - Use `<attr name=type" type="string">Category_Kind</attr>` to define a category type such as personal or corporate.

    > **Note**
    >
    > The "type" attribute can be populated with one category, either "Personal" or "Corporate". "Personal" is taken as the default.

  - Use `<attr name="parentId" type="string">parent_Id</attr>` to define the parent ID.

Response:

- Body:
  The response body contains the details of category created .

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">demo1</title>
    <updated>2016-05-17T08:59:03.748Z</updated>
    <link href="http://localhost:6405/biprws/v1/categories/6406"/>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">AT04yiTr6ixMsQ.XbCysgWA</attr>
            <attr name="keywords" type="string"></attr>
            <attr name="parentcuid"
type="string">CQEvAAAA9FZpPn7XfU2Ust.ltEwspg</attr>
```

```
            <attr name="created" type="datetime">2016-05-17T08:59:03.725Z</attr>
            <attr name="name" type="string">demo1</attr>
            <attr name="description" type="string"></attr>
            <attr name="id" type="string">6406</attr>
            <attr name="ownerid" type="string">12</attr>
            <attr name="type" type="string">PersonalCategory</attr>
            <attr name="updated" type="string">Tue May 17 01:59:03 PDT 2016</
attr>
            <attr name="parentid" type="string">889</attr>
        </attrs>
    </content>
</entry>
```

7.12.3 Getting Category Details

GET http://<baseURL>/v1/categories/<category_Id>

Get the details of a category using the `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/categories/<category_Id>`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Body:
  An XML entry of category details is displayed.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">category</title>
    <updated>2016-05-17T09:08:47.037Z</updated>
    <link href="http://localhost:6405/biprws/v1/categories/6406"/>
    <link href="http://localhost:6405/biprws/v1/categories/6406/children"/>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">AT04yiTr6ixMsQ.XbCysgWA</attr>
            <attr name="keywords" type="string"></attr>
            <attr name="parentcuid"
type="string">CQEvAAAA9FZpPn7XfU2Ust.ltEwspg</attr>
            <attr name="created" type="datetime">2016-05-17T08:59:03.725Z</attr>
            <attr name="name" type="string">demo1</attr>
            <attr name="description" type="string"></attr>
```

```
            <attr name="id" type="string">6406</attr>
            <attr name="ownerid" type="string">12</attr>
            <attr name="type" type="string">PersonalCategory</attr>
            <attr name="updated" type="string">Tue May 17 01:59:03 PDT 2016</
attr>
            <attr name="parentid" type="string">889</attr>
        </attrs>
    </content>
</entry>
```

7.12.4 Modifying Category Details

PUT http://<baseURL>/v1/category/<category_ID>

To modify a category, use the `PUT` request to `/v1/categories/category_Id>` URL.

Request:

- Method: `PUT`
- URL: `http://<baseURL>/v1/categories/<category_Id>`
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">category</title>
    <updated>2016-05-17T09:08:47.037Z</updated>
    <link href="http://10.160.205.83:6405/biprws/v1/categories/6406"/>
    <link href="http://10.160.205.83:6405/biprws/v1/categories/6406/children"/>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">AT04yiTr6ixMsQ.XbCysgWA</attr>
            <attr name="keywords" type="string"></attr>
            <attr name="parentcuid"
type="string">CQEvAAAA9FZpPn7XfU2Ust.ltEwspg</attr>
            <attr name="created" type="datetime">2016-05-17T08:59:03.725Z</attr>
            <attr name="name" type="string">Test_Category</attr>
            <attr name="description" type="string"></attr>
            <attr name="id" type="string">6406</attr>
            <attr name="ownerid" type="string">12</attr>
            <attr name="type" type="string">PersonalCategory</attr>
            <attr name="updated" type="string">Tue May 17 01:59:03 PDT 2016</
attr>
            <attr name="parentid" type="string">889</attr>
        </attrs>
    </content>
</entry>
```

- Use `<attr name="Keyword" type="string">Modified</attr>` to define the keyword for the category.
- Use `<attr name="name" type="string">Test_Category</attr>` to define to define the title of the category.
- Use `<attr name="description" type="string">Modified Category</attr>` to define the description of the category.
- Use `<attr name="ownerid" type="string">12</attr>` to define the description of the category.
- Use `<attr name="type" type="string">CorporateCategory</attr>` to define the category type.
- Use `<attr name="ParentId" type="string">889</antr>` to define the parent ID of the category.

Response:

- Body:
  The response body contains the details of the modified category.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">category</title>
    <updated>2016-05-17T09:08:47.037Z</updated>
    <link href="http://10.160.205.83:6405/biprws/v1/categories/6406"/>
    <link href="http://10.160.205.83:6405/biprws/v1/categories/6406/children"/>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">AT04yiTr6ixMsQ.XbCysgWA</attr>
            <attr name="keywords" type="string">Modified</attr>
            <attr name="parentcuid"
type="string">CQEvAAAA9FZpPn7XfU2Ust.ltEwspg</attr>
            <attr name="created" type="datetime">2016-05-17T08:59:03.725Z</attr>
            <attr name="name" type="string">Test_Category</attr>
            <attr name="description" type="string">Modified Category</attr>
            <attr name="id" type="string">6406</attr>
            <attr name="ownerid" type="string">12</attr>
            <attr name="type" type="string">CorporateCategory</attr>
            <attr name="updated" type="string">Tue May 17 01:59:03 PDT 2016</
attr>
            <attr name="parentid" type="string">889</attr>
        </attrs>
    </content>
</entry>
```

7.12.5 Listing First Level Childern Under a Category

GET http://<baseURL>/v1/categories/<parentcategory_Id>/children

Get the list of first level children under a category using the `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/categories/<parentcategory_Id>/children`

- Header:

Name	Value
Content-Type	application/xml
Accept	application/xml
X-SAP-LogonToken	The logon token value, in quotation marks.

- Body: none

Response:

- Body:
  An XML feed of the first level children under a category created in the BOE system is displayed.


```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">category</title>
    <updated>2016-05-17T09:18:03.096Z</updated>
    <entry>
        <title type="text">demomodified</title>
        <updated>2016-05-17T09:18:03.518Z</updated>
        <link href="http://localhost:6405/biprws/v1/categories/889/children"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AT04yiTr6ixMsQ.XbCysgWA</attr>
                <attr name="keywords" type="string"></attr>
                <attr name="created" type="string"></attr>
                <attr name="name" type="string">demomodified</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">6406</attr>
                <attr name="type" type="string">PersonalCategory</attr>
                <attr name="updated" type="string" null="true"/>
                <attr name="parentid" type="string">889</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">Fold Unfold Sample</title>
        <link href="http://localhost:6405/biprws/v1/categories/889/children"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AXpyoryP_Z5BoXVoM2_3kWQ</attr>
                <attr name="keywords" type="string"></attr>
                <attr name="created" type="string"></attr>
                <attr name="name" type="string">Fold Unfold Sample</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">5734</attr>
                <attr name="type" type="string">Webi</attr>
                <attr name="updated" type="string" null="true"/>
                <attr name="parentid" type="string">5712</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">Formatting Sample</title>
        <link href="http://localhost:6405/biprws/v1/categories/889/children"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AQtkbbSqN4NOj3ydf.Sw1lY</attr>
                <attr name="keywords" type="string"></attr>
                <attr name="created" type="string"></attr>
                <attr name="name" type="string">Formatting Sample</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">5730</attr>
```

```
                <attr name="type" type="string">Webi</attr>
                <attr name="updated" type="string" null="true"/>
                <attr name="parentid" type="string">5712</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">vg1</title>
        <updated>2016-05-17T09:18:03.533Z</updated>
        <link href="http://localhost:6405/biprws/v1/categories/889/children"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AfoVLN1zVt5AqaP5YabWlM0</attr>
                <attr name="keywords" type="string"></attr>
                <attr name="created" type="string"></attr>
                <attr name="name" type="string">vg1</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">6411</attr>
                <attr name="type" type="string">PersonalCategory</attr>
                <attr name="updated" type="string" null="true"/>
                <attr name="parentid" type="string">889</antr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.12.6 Listing the First Level Documents in a category

GET http://<baseURL>/v1/categories/<parentcategory_Id>/documents

Get the list of first level children under a category using the `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/categories/<parentcategory_Id>/documents`
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Body:
  An XML feed of the first level children under a category created in the BOE system is displayed.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">category</title>
```

```
    <updated>2016-05-17T09:18:50.985Z</updated>
    <entry>
        <title type="text">Fold Unfold Sample</title>
        <updated>2016-05-17T09:18:50.998Z</updated>
        <link href="http://10.160.205.83:6405/biprws/v1/categories/889/
documents"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AXpyoryP_Z5BoXVoM2_3kWQ</attr>
                <attr name="keywords" type="string"></attr>
                <attr name="created" type="string"></attr>
                <attr name="name" type="string">Fold Unfold Sample</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">5734</attr>
                <attr name="type" type="string">Webi</attr>
                <attr name="updated" type="string" null="true"/>
                <attr name="parentid" type="string">5712</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">Formatting Sample</title>
        <updated>2016-05-17T09:18:51.001Z</updated>
        <link href="http://10.160.205.83:6405/biprws/v1/categories/889/
documents"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AQtkbbSqN4NOj3ydf.Sw1lY</attr>
                <attr name="keywords" type="string"></attr>
                <attr name="created" type="string"></attr>
                <attr name="name" type="string">Formatting Sample</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">5730</attr>
                <attr name="type" type="string">Webi</attr>
                <attr name="updated" type="string" null="true"/>
                <attr name="parentid" type="string">5712</antr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.12.7 Deleting Categories

DELETE http://<baseURL>/v1/categories/<categoryID>

Delete a category using the `DELETE` method.

Request:

- Method: DELETE
- URL: `DELETE http://<baseURL>/v1/categories/<categoryID>`
- Header:

Name	Value
Content-Type	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Body:
  An XML feed displays either a success or failure message for the delete operation. This example shows successful deletion of a category.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">category</title>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="Status" type="string">Success</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.13 Managing Folders

This section provides you information on RESTful APIs to manage folders. You can use these APIs to perform the following activities:

- List folders
- Create a folder
- Get details of a folder
- Modify details of a folder
- List the first level children under a folder
- Upload a file to the folder
- Delete a folder

The following convention applies to the REST APIs described in document management:

`http://<baseURL>/<vx>/folders`

Where <vx>=v1

7.13.1 Listing Folders

GET http://<baseURL>/v1/folders

Use the `GET` method to list the folders.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/folders`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Body:
  You can see an XML feed of all the root folders created in the BOE system.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Folder</title>
    <updated>2016-06-11T16:43:25.207Z</updated>
    <link href="http://localhost:6405/biprws/v1/folders?page=1&amp;pagesize=50"
rel="self"/>
    <link href="http://localhost:6405/biprws/v1/folders?page=1&amp;pagesize=50"
rel="first"/>
    <link href="http://localhost:6405/biprws/v1/folders?page=1&amp;pagesize=50"
rel="last"/>
    <entry>
        <title type="text">Auditing</title>
        <link href="http://localhost:6405/biprws/v1/folders/585"
rel="alternate"/>
        <updated>2016-05-17T11:28:26.058Z</updated>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AS1oZEJAynpNjZIaZK2rc7g</attr>
                <attr name="name" type="string">Auditing</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">585</attr>
                <attr name="type" type="string">Folder</attr>
                <attr name="ownerid" type="string">10</attr>
                <attr name="updated" type="string">Tue May 17 04:28:26 PDT 2016</
attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">Data Federation</title>
```

```
        <link href="http://localhost:6405/biprws/v1/folders/4695"
rel="alternate"/>
        <updated>2016-05-17T11:36:43.699Z</updated>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid"
type="string">FnKsrkkctAcA8BAAALB7kkQAADAFzVMX</attr>
                <attr name="name" type="string">Data Federation</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">4695</attr>
                <attr name="type" type="string">Folder</attr>
                <attr name="ownerid" type="string">10</attr>
                <attr name="updated" type="string">Tue May 17 04:36:43 PDT 2016</
attr>
            </attrs>
        </content>
    </entry>
    <entry>
</feed>
```

Sorting folder

Sorting is the process of arranging the folders in a systematic order.

You can now sort folders sequentially in ascending or descending order based on the following attributes:

- Name
- Description
- ID of the folder
- Type

Filtering folders

You can filter folders based on the *type* and *updated* attribute.

The query parameters for sort and filter is as follows:

Query Parameter

Parameter Name	Parameter Value
Sort	`sort=<+/-><attribute>` **Note** • For ascending order, the value is + • For descending order, the value is - • By default, the folders are sorted in ascending based on name of the folder **Example** sort=+name
Filter	`type=<folder>`

Parameter Name	Parameter Value
Filter	`updated=<Start date year-month-dateThh:mm:ss.Timezone> , <End date year-month-dateThh:mm:ss.Timezone>` **Note** If no end date is specified, the current date is taken. **Example** updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z
Paging	• `page=<page number>` • `pagesize=<number>` `pagesize` is the number of objects displayed in the response **Note** By default, `page=1` and `pagesize=50` that is, in one page number 1, the top 50 objects will be displayed in the response.

URL: `GET http://<baseURL>/v1/folders?sort=<name>&updated=<Start year-month-dateThh:mm:ss.727Z> , <End year-month-dateThh:mm:ss.727Z>&page=<page number>&pagesize=<number>`

Example

URL: `GET http://<baseURL>/v1/folders?sort=+name&updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z&page=2&pagesize=5`

Once you send the request, the response displays the second page, which has five objects and the list of folders updated between the specified date and sorted in ascending order.

7.13.2 Creating a Folder

Make a request to the `v1/folders/folder` URL to create a folder in the BI platform.

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to create a folder in the BI platform.

GET http://<baseURL>/v1/folders/folder

Make a `GET` request to `v1/folders/folder` to receive a XML template. Use the XML template in the request body of the `POST` method to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/folders/folder`
- Header:

Name	Value
`Content Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="keywords" type="string"></attr>
            <attr name="name" type="string"></attr>
            <attr name="description" type="string"></attr>
            <attr name="parentid" type="string"></attr>
        </attrs>
    </content>
</entry>
```

POST http://<baseURL>/v1/folders/folder

To create a folder, select the `POST` method and request to `v1/folders/folder` URL.

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/folders/folder`
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
```

```
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="keywords" type="string"></attr>
            <attr name="name" type="string"></attr>
            <attr name="description" type="string"></attr>
            <attr name="parentid" type="string"></attr>
        </attrs>
    </content>
</entry>
```

- Use `<attr name="keywords" type="string">folder_keywords</attr>` to define keywords for the folder.
- Use `<attr name="name" type="string">folder_Name</attr>` to define folder name.
- Use `<attr name="description" type="string">folder_description</attr>` to enter folder description.
- Use `<attr name="Parentid" type="string">Parent_ID</attr>` to define parent ID.

Response:

- Body:
  The response body contains the details of the folder that you have created .

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/infostore/12</uri>
    </author>
    <title type="text">Folder_New</title>
    <updated>2016-03-14T08:08:43.417Z</updated>
    <link href="http://localhost:6405/biprws/v1/folders/6565/children"
rel="alternate" />
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">ARMpb.GxruhFlMx..hxliAM</attr>
            <attr name="keywords" type="string">Creating Folder</attr>
            <attr name="parentcuid" type="string">AS1oZEJAynpNjZIaZK2rc7g</attr>
            <attr name="name" type="string">Folder_New</attr>
            <attr name="description" type="string">Testing Folder Creation</attr>
            <attr name="id" type="int32">6565</attr>
            <attr name="parentid" type="int32">522</attr>
        </attrs>
    </content>
</entry>
```

7.13.3 Getting Folder Details

GET http://<baseURL>/v1/folders/<folder_id>

Use the `GET` method to retrieve the folder details.

Request:

- Method: GET

- URL: `GET http://<baseURL>/v1/folders/<folder_id>`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Body:
  You can see the folder details in an XML template.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Folder</title>
    <updated>2016-03-14T08:26:39.289Z</updated>
    <id>tag:sap.com,2010:bip-rs/ARMpb.GxruhFlMx..hxliAM</id>
    <link href="http://localhost:6405/biprws/v1/folders/6565folders/6565/
children" rel="alternate" />
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">ARMpb.GxruhFlMx..hxliAM</attr>
            <attr name="keywords" type="string">Creating Folder</attr>
            <attr name="created" type="string">Mon Mar 14 09:08:43 CET 2016</
attr>
            <attr name="parentcuid" type="string">AS1oZEJAynpNjZIaZK2rc7g</attr>
            <attr name="name" type="string">Folder_New</attr>
            <attr name="description" type="string">Testing Folder Creation</attr>
            <attr name="id" type="int32">6565</attr>
            <attr name="type" type="string">Folder</attr>
            <attr name="ownerid" type="int32">12</attr>
            <attr name="updated" type="string">Mon Mar 14 09:08:43 CET 2016</
attr>
            <attr name="parentid" type="int32">522</attr>
        </attrs>
    </content>
</entry>
```

7.13.4 Modifying folder details

PUT http://<baseURL>/v1/folders/<folder_ID>

Use `PUT` request to `/v1/folders/folder_Id>` URL to modify folder details.

Request:

- Method: `PUT`
- URL: `http://<baseURL>/v1/folders/<folder_Id>`

- Header:

Name	Value
Content-Type	application/xml
Accept	application/xml
X-SAP-LogonToken	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Folder</title>
    <updated>2016-06-11T16:50:29.554Z</updated>
    <link href="http://localhost:6405/biprws/v1/folders/585" rel="alternate"/>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">AS1oZEJAynpNjZIaZK2rc7g</attr>
            <attr name="keywords" type="string"></attr>
            <attr name="created" type="string">Tue May 17 04:28:09 PDT 2016</
attr>
            <attr name="parentcuid" type="string">ASHnC0S_Pw5LhKFbZ.iA_j4</attr>
            <attr name="name" type="string">Auditing</attr>
            <attr name="description" type="string"></attr>
            <attr name="id" type="string">585</attr>
            <attr name="type" type="string">Folder</attr>
            <attr name="ownerid" type="string">10</attr>
            <attr name="updated" type="string">Tue May 17 04:28:26 PDT 2016</
attr>
            <attr name="parentid" type="string">23</attr>
        </attrs>
    </content>
</entry>
```

  - Use `<attr name="keywords" type="string">Modified</attr>` to define keywords for the folder.
  - Use `<attr name="name" type="string">Creating Folder</attr>` to define the name of the folder.
  - Use `<attr name="description" type="string">Modified Folder</attr>` to define a description for the folder.
  - Use `<attr name="parentid" type="int32">518</attr>` to define the parent ID of the folder.

Response:

- Body:
  The response body contains the details of modified folder.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Folder</title>
    <updated>2016-06-11T16:50:53.559Z</updated>
    <link href="http://localhost:6405/biprws/v1/folders/585" rel="alternate"/>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">AS1oZEJAynpNjZIaZK2rc7g</attr>
            <attr name="keywords" type="string"></attr>
            <attr name="created" type="string">Tue May 17 04:28:09 PDT 2016</
attr>
            <attr name="parentcuid" type="string">ASHnC0S_Pw5LhKFbZ.iA_j4</attr>
            <attr name="name" type="string">Auditing1</attr>
            <attr name="description" type="string"></attr>
            <attr name="id" type="string">585</attr>
            <attr name="type" type="string">Folder</attr>
            <attr name="ownerid" type="string">10</attr>
```

```
            <attr name="updated" type="string">Tue May 17 04:28:26 PDT 2016</
attr>
            <attr name="parentid" type="string">23</attr>
        </attrs>
    </content>
</entry>
```

7.13.5 Listing First Level Children Under a Folder

GET http://<baseURL>/v1/folders/parentfolder_id/folder

Use the `GET` method to get the list of first level children under a folder.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/folders/parentfolder_id/children`
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Body:
  An XML feed of the first level children under a folder created in BOE system is displayed.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Folder</title>
    <updated>2016-03-15T11:01:23.586Z</updated>
    <link href="http://localhost:6405/biprws/v1/folders/23/children?
page=1&amp;pagesize=50" rel="self" />
    <link href="http://localhost:6405/biprws/v1/folders/23/children?
page=1&amp;pagesize=50" rel="first" />
    <link href="http://localhost:6405/biprws/v1/folders/23/children?
page=1&amp;pagesize=50" rel="last" />
    <entry>
        <title type="text">Auditing</title>
        <link href="http://localhost:6405/biprws/v1/folders/23/children"
rel="alternate" />
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AS1oZEJAynpNjZIaZK2rc7g</attr>
                <attr name="name" type="string">Auditing</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">522</attr>
                <attr name="type" type="string">Folder</attr>
                <attr name="ownerid" type="string">10</attr>
```

```
                <attr name="updated" type="string">Fri Mar 04 17:24:25 CET 2016</
attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.13.6 Uploading File to Folder

http://<baseURL>/biprws/v1/folders/<folder_id>/file

Make a `POST` request using `/v1/folders/<folder_id/file` to upload a file.

Request:

1. Method: `POST`
2. URL: `http://<baseURL>/v1/folders/<folder_id>/file`
   Replace `<baseURL>` with the base URL for RESTful web service requests. Replace `<folder_id>` with the ID of the folder where you want to upload a file to.
3. Header:

Name	Value
`X-SAP-LogonToken`	The logon token value, in quotation marks.

4. In the *Body* tab, select the *form data* radio button.

   > **Note**
   >
   > The workflow is executed in the Postman REST client

5. Enter any keyword and select *Choose Files*.

   > **Note**
   >
   > The workflow is executed in the Postman REST client

> **Note**
>
> - You can upload any BOE or non-BOE documents.
> - You can also upload multiple BOE and non -BOE documents at the same time.
> - BOE files: Webi, Crystal and Lumira documents are uploaded as *Web Intelligence*, *Crystal Reports* and *Lumira Documents* respectively. Apart from these BOE docuemnts, other documents are uploaded as *Agnostic*.
> - Non-BOE files include word, pdf, and txt. These documents are uploaded as *Word Document*, *Adobe Acrobat* and *Text* respectively.

Response:

- Body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <updated>2016-05-24T11:23:02.624Z</updated>
    <entry>
        <title type="text">crashlog</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AbR0epv8S4tNoR5skcwwZYg</attr>
                <attr name="name" type="string">crashlog</attr>
                <attr name="id" type="string">7583</attr>
                <attr name="type" type="string">Txt</attr>
                <attr name="parentid" type="string">5712</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

7.13.7 Deleting a Folder

DELETE http://<baseURL>/v1/folders/<folder_id>

Use the `DELETE` method to delete a folder.

Request:

- Method: DELETE
- URL: `DELETE http://<baseURL>/v1/folders/<folder_id>`
- Header:

Name	Value
Content-Type	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Body:
  An XML template displays eithe a success or failure message for the delete operation. This example shows successful deletion of a folder.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="status" type="string">success</attr>
        </attrs>
    </content>
```

```
</entry>
```

7.14 Managing Documents

This section provides you with information about RESTful APIs to manage documents. Using these APIs, you can perform the following tasks:

- List documents
- Get details of a document
- Modify details of a document
- Get the category details of a document
- Delete a document

The following conventions apply to the REST APIs described in document management:

`http://<baseURL>/<vx>/documents`

Where <vx>=v1

7.14.1 Listing Documents

GET http://<baseURL>/v1/documents

Use the `GET` method to list all the documents.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/documents`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Body:

An XML feed of all the documents present in the BOE system is listed.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <id>tag:sap.com,2010:bip-rs/infostore</id>
    <title type="text">document</title>
    <updated>2016-03-17T08:34:32.976Z</updated>
    <link href="http://10.208.38.216:6405/biprws/v1/documents?
page=1&amp;pagesize=50" rel="self" />
    <link href="http://10.208.38.216:6405/biprws/v1/documents?
page=1&amp;pagesize=50" rel="first" />
    <link href="http://10.208.38.216:6405/biprws/v1/documents?
page=1&amp;pagesize=50" rel="last" />
    <entry>
        <link href="http://10.208.38.216:6405/biprws/documents/7057" />
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AecjVLlZDstJiUcCwmRrBMo</attr>
                <attr name="name" type="string">Test1</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">7057</attr>
                <attr name="type" type="string">Txt</attr>
                <attr name="ownerid" type="string">12</attr>
                <attr name="updated" type="string">Wed Mar 16 10:33:43 CET 2016</
attr>
                <attr name="parentid" type="string">5330</attr>
            </attrs>
        </content>
    </entry>
    <entry>
</feed>
```

Sorting documents

Sorting is the process of arranging the documents in a systematic order.

You can now sort documents sequentially in ascending or descending order based on the following attributes:

- Name
- Description
- ID of the document
- Type

Filtering documents

Filtering documents is the process of filtering the children under a particular hierarchy. You can filter documents based on the *type* and *updated* attribute.

The query parameters for sort and filter are as follows:

Query Parameter

Parameter Name	Parameter Value
Sort	`sort=<+><attribute>` **Note** • For ascending order, the value is + • For descending order, the value is - • By default, the documents are sorted in ascending order based on the name of the document **Example** sort=+name
Filter	`type=<document_type>`
Filter	`updated=<Start date year-month-dateThh:mm:ss.Timezone> , <End date year-month-dateThh:mm:ss.Timezone>` **Note** If no end date is specified, the current date will be taken. **Example** updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z
Paging	• `page=<page number>` • `pagesize=<number>` `pagesize` is the number of objects displayed in the response **Note** By default `page=1` and `pagesize=50` that is, in one page number 1, the top 50 objects will be displayed in the response.

URL: `GET http://<baseURL>/v1/documents?sort=<name>&updated=<Start year-month-dateThh:mm:ss.727Z> , <End year-month-dateThh:mm:ss.727Z>&page=<page number>&pagesize=<number>`

Example

URL: `GET http://<baseURL>/v1/documents?sort=+name&updated=2015-11-16T11:03:53.727Z , 2016-11-16T11:03:53.727Z&page=2&pagesize=5`

Once you send the request, the response displays the second page, which has five objects and the list of documents updated between the specified date and sorted in ascending order.

7.14.2 Getting Document Details and the OpenDoc URL

GET http://<baseURL>/v1/documents/<document_id>

Use the `GET` method to get the details of a document.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/documents/<document_id>`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Body:
  The details of a document are displayed in an XML entry.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">document</title>
    <updated>2016-03-17T08:37:01.354Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">AecjVLlZDstJiUcCwmRrBMo</attr>
            <attr name="keywords" type="string"></attr>
            <attr name="parentcuid" type="string">AeN4lEu0h_tAtnPEjFYxwi8</attr>
            <attr name="created" type="string">Wed Mar 16 10:01:38 CET 2016</
attr>
            <attr name="name" type="string">Test1</attr>
            <attr name="description" type="string"></attr>
            <attr name="id" type="string">7057</attr>
            <attr name="type" type="string">Txt</attr>
            <attr name="ownerid" type="string">12</attr>
            <attr name="updated" type="string">Wed Mar 16 10:33:43 CET 2016</
attr>
            <attr name="parentid" type="string">5330</attr>
        </attrs>
    </content>
</entry>
```

> **Note**
>
> If you pass the **query parameter** `<details>` **with a value of 'true'**, the server also provides you with additional document details including the **OpenDoc URL** of the document. This is shown in the below example:
>
> Server response for URL: `http://localhost:6405/biprws/v1/documents/24025?details=true` used with the GET method:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">document</title>
    <updated>2016-03-17T08:37:01.354Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
          <attr name="owner" type="string">AecjVLlZDstJiUcCwmRrBMo</attr>
          <attr name="docLink" type="string">http://TargetMachine:8080/BOE/
OpenDocument/opendoc/openDocument.jsp?
sIDType=CUID&iDocID=ATAhWqo5J_NIiS9o_90S6F4</attr>
            <attr name="cuid" type="string">AecjVLlZDstJiUcCwmRrBMo</attr>
            <attr name="keywords" type="string"></attr>
            <attr name="parentcuid" type="string">AeN4lEu0h_tAtnPEjFYxwi8</attr>
            <attr name="created" type="string">Wed Mar 16 10:01:38 CET 2016</attr>
            <attr name="name" type="string">Calculated Member Cross-tab Chart:
24025</attr>
            <attr name="description" type="string"></attr>
          <attr name="locale" type="string"></attr>
            <attr name="id" type="string">7057</attr>
            <attr name="type" type="string">Txt</attr>
            <attr name="ownerid" type="string">12</attr>
            <attr name="updated" type="string">Wed Mar 16 10:33:43 CET 2016</attr>
            <attr name="parentid" type="string">5330</attr>
        </attrs>
    </content>
</entry>
```

7.14.3 Modifying Document Details

PUT http://<baseURL>/v1/documents/<document_id>

To modify a document, use `PUT` request to `/v1/documents/<document_id>` URL.

Request:

- Method: `PUT`
- URL: `http://<baseURL>/v1/documents/<document_id>`
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`

Name	Value
X-SAP-LogonToken	The logon token value, in quotation marks.

- Body:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">document</title>
    <updated>2016-03-16T09:36:14.523Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">AecjVLlZDstJiUcCwmRrBMo</attr>
            <attr name="keywords" type="string"></attr>
            <attr name="parentcuid" type="string">AeN4lEu0h_tAtnPEjFYxwi8</attr>
            <attr name="created" type="string">Wed Mar 16 10:01:38 CET 2016</
attr>
            <attr name="name" type="string">Test1</attr>
            <attr name="description" type="string"></attr>
            <attr name="id" type="string">7057</attr>
            <attr name="type" type="string">Txt</attr>
            <attr name="ownerid" type="string">12</attr>
            <attr name="updated" type="string">Wed Mar 16 10:33:43 CET 2016</
attr>
            <attr name="parentid" type="string">5330</attr>
        </attrs>
    </content>
</entry>
```

  - Use `<attr name="keywords" type="string">Modified</attr>` to define keyword for the document.
  - Use `<attr name="name" type="string">Creating document</attr>` to define the name of the document.
  - Use `<attr name="description" type="string">Modified document</attr>` to define description of the document.
  - Use `<attr name="parentid" type="string">518</attr>` to define parent ID of the document.

Response:

- Body:
  The response body contains the details of the modified document.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <title type="text">document</title>
    <updated>2016-03-16T09:36:14.523Z</updated>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">AecjVLlZDstJiUcCwmRrBMo</attr>
            <attr name="keywords" type="string"></attr>
            <attr name="parentcuid" type="string">AeN4lEu0h_tAtnPEjFYxwi8</attr>
            <attr name="created" type="string">Wed Mar 16 10:01:38 CET 2016</
attr>
            <attr name="name" type="string">Test2</attr>
            <attr name="description" type="string"></attr>
            <attr name="id" type="string">7057</attr>
            <attr name="type" type="string">Txt</attr>
            <attr name="ownerid" type="string">12</attr>
            <attr name="updated" type="string">Wed Mar 16 10:33:43 CET 2016</
attr>
            <attr name="parentid" type="string">5330</attr>
        </attrs>
    </content>
</entry>
```

7.14.4 Getting Category Details of Document

GET http://<baseURL>/v1/documents/<document_id>/categories

Get the category details of a document using the `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/documents/<document_id>/categories`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Body:
  An XML feed of category details of a document is displayed.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">CQEvAAAA9FZpPn7XfU2Ust.ltEwspg</
attr>
                <attr name="name" type="string">Administrator</attr>
                <attr name="id" type="string">814</attr>
                <attr name="type" type="string">PersonalCategory</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AQkGptd3yOhIuOmUSJDe8g4</attr>
                <attr name="name" type="string">Test</attr>
                <attr name="id" type="string">8088</attr>
                <attr name="type" type="string">Category</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">ARTW.R1U1BdBhb.pl3I0f0I</attr>
                <attr name="name" type="string">CorporateCategory</attr>
                <attr name="id" type="string">8089</attr>
                <attr name="type" type="string">Category</attr>
            </attrs>
```

```
        </content>
    </entry>
</feed>
</entry>
```

7.14.5 Assigning a Document to Categories

You can assign a document to a single or multiple categories.

POST `http://<base URL>/v1/categories/<category_id>/documents`

- Use the `POST` method using the following details:
  - URL: `http://<base URL>/v1/categories/<category_id>/documents`

    > **Note**
    >
    > `<category_id>` is the category ID to which you want to assign the documents. To fetch the valid categories available on the server, see the related topic.

  - Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

  - Body:

```
feed xmlns="http://www.w3.org/2005/Atom">
    <entry>

        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="id" type="string">7278</attr>
          </attrs>
        </content>
    </entry>
</feed>
```

    TASK-RESULT: The specified documents are assigned to the category ID passed in the URL.The server indicates the status with a "success" message.

Related Information

Listing categories [page 262]

7.14.6 Removing Categories from a Document

You can remove the categories assigned to BI documents.

DELETE `http://<base URL>/v1/categories/<category_id>/documents`

- Use the `DELETE` method using the following details:
  - URL: `http://<base URL>/v1/categories/<category_id>/documents`

    > **Note**
    >
    > `<category_id>` is the category ID from which you want to remove the documents. **To fetch the category details of a document, see the related topic**.

  - Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

  - Body:

```
feed xmlns="http://www.w3.org/2005/Atom">
    <entry>

        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="id" type="string">7278</attr>
          </attrs>
        </content>
    </entry>
</feed>
```

    TASK-RESULT: The specified documents are removed from the category ID passed in the URL. The server indicates the status with a "success" message.

Related Information

Getting Category Details of Document [page 290]

7.14.7 Sending Documents To Various Destinations

Using RESTful web service APIs, you can send a document to a file destination, a user's mailbox (SMTP), to an FTP location or to the BI Inbox.

a. Sending Documents to a User's BI Inbox

GET `http://<baseURL>/v1/documents/destinations/inbox`

- Use the `GET` method to fetch the request body using the following details:
  - URL: `http://<baseURL>/v1/documents/destinations/inbox`

    > **Note**
    >
    > Replace `<baseURL>` with the base URL for RESTful web service requests (such as `host:<port>/biprws`).

  - Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

  - Body: none
    The request template is received from the server:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <title type="text">SendToInbox</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="docid" type="string"></attr>
                <attr name="usergroupid" type="string"></attr>
                <attr name="userid" type="string"></attr>
            </attrs>
        </content>
    </entry>
</feed>
```

POST `http://<baseURL>/v1/documents/destinations/inbox`

- Use the `POST` method to post the send request & pass your specific details in the body:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
```

```
        <title type="text">SendToInbox</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="docid" type="string">6754</attr>
                <attr name="usergroupid" type="string"></attr>
                <attr name="userid" type="string">albert.doughlas@sap.com</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

Note

- If you want to everytime send a specific document using the default request body, you can use the query parameter `'usedefaults'` (e.g `http://host:<port>/biprws/vx/documents/destinations/inbox?usedefaults=true`). You recieve the following request body that you can post with the `docid`:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <title type="text">SendToInbox</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="docid" type="string"></attr>
            </attrs>
        </content>
    </entry>
</feed>
```

- Other optional query parameters are `?targetname` and `?shortcut`.

Task result: The specific document is sent to the specified users.

b. Sending Documents to a Disk Location

GET `http://<baseURL>/v1/documents/destinations/disk`

- Use the `GET` method to fetch the request body using the following details:
  - URL: `http://<baseURL>/v1/documents/destinations/disk`

    Note

    Replace `<baseURL>` with the base URL for RESTful web service requests (such as `host:<port>/biprws`).

  - Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`

Name	Value
X-SAP-LogonToken	The logon token value, in quotation marks.

- Body: none
  The following response is received from the server:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <title type="text">SendToFileSystem</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="password" type="string"></attr>
                <attr name="docid" type="string"></attr>
                <attr name="targetobjectname" type="string"></attr>
                <attr name="directory" type="string">.</attr>
                <attr name="username" type="string"></attr>
            </attrs>
        </content>
    </entry>
</feed>
```

POST http://<baseURL>/v1/documents/destinations/disk

- Use the POST method to post the send request using the same URL and header, but the body with your specific details:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <title type="text">SendToFileSystem</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="password" type="string">bobobo</attr>
                <attr name="docid" type="string">5458</attr>
                <attr name="targetobjectname" type="string"></attr>
                <attr name="directory" type="string">.C:\</attr>
                <attr name="username" type="string">administrator</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

Note

- If you want to use the default request body everytime, you can use the query parameter 'usedefaults' (e.g http://host:<port>/biprws/vx/documents/destinations/disk?usedefaults=true). You recieve the following request body that you can post with the docid:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <title type="text">SendToFileSystem</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="docid" type="string"></attr>
            </attrs>
        </content>
    </entry>
</feed>
```

Task result: The specific document is sent to the specified disk location.

c. Sending Documents Via Email (SMTP)

GET `http://<baseURL>/v1/documents/destinations/smtp`

- Use the `GET` method to fetch the request body using the following details:
  - URL: `http://<baseURL>/v1/documents/destinations/smtp`

    > **Note**
    >
    > Replace `<baseURL>` with the base URL for RESTful web service requests (such as `host:<port>/biprws`).

  - Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

  - Body: none
    The following request template is received from the server:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <title type="text">SendToEmail</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cc" type="string"></attr>
                <attr name="bcc" type="string"></attr>
                <attr name="docid" type="string"></attr>
                <attr name="subject" type="string"></attr>
                <attr name="from" type="string"></attr>
                <attr name="to" type="string"></attr>
                <attr name="useattachment" type="bool">true</attr>
                <attr name="message" type="string"></attr>
                <attr name="targetObjectName" type="string"></attr>
            </attrs>
        </content>
    </entry>
</feed>
```

POST `http://<baseURL>/v1/documents/destinations/smtp`

- Use the `POST` method to post the send request using the same URL and header, but the body with your specific details:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <title type="text">SendToEmail</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cc" type="string">g.garg@sap.com</attr>
                <attr name="bcc" type="string"></attr>
                <attr name="docid" type="string">5438</attr>
                <attr name="subject" type="string">Sales Reports</attr>
                <attr name="from" type="string">bob.walter@sap.com</attr>
                <attr name="to" type="string">albert.waterman@sap.com</attr>
                <attr name="useattachment" type="bool">true</attr>
                <attr name="message" type="string">Find my reports attached
please!</attr>
                <attr name="targetObjectName" type="string"></attr>
            </attrs>
        </content>
    </entry>
</feed>
```

Note

- If you want to everytime send a specific document using the default request body, you can use the query parameter `'usedefaults'` (e.g `http://host:<port>/biprws/vx/documents/destinations/smtp?usedefaults=true`). You recieve the following request body that you can post with the `docid`:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <title type="text">SendToEmail</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="docid" type="string"></attr>
            </attrs>
        </content>
    </entry>
</feed>
```

- Configure the following in the Central Management Console (CMC) for SMTP:
  - Domain Name
  - Host
  - Port

Task result: The specific document is emailed based on the attribute values passed on the body.

d. Sending Documents to an FTP location

GET `http://<baseURL>/v1/documents/destinations/ftp`

- Use the `GET` method to fetch the request body using the following details:
  - URL: `http://<baseURL>/v1/documents/destinations/ftp`

    > **Note**
    >
    > Replace `<baseURL>` with the base URL for RESTful web service requests (such as `host:<port>/biprws`).

  - Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

  - Body: none
    The following request template is received from the server:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <title type="text">SendToFtp</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="password" type="string"></attr>
                <attr name="port" type="int32">21</attr>
                <attr name="docid" type="string"></attr>
                <attr name="host" type="string"></attr>
                <attr name="targetobjectname" type="string"></attr>
                <attr name="directory" type="string">.</attr>
                <attr name="account" type="string"></attr>
                <attr name="username" type="string"></attr>
            </attrs>
        </content>
    </entry>
</feed>
```

POST `http://<baseURL>/v1/documents/destinations/ftp`

- Use the `POST` method to post the send request using the same URL and header, but the body with your specific details.
  Task result: The specific document is sent to the FTP location based on the attribute values passed on the body.

e. Sending Documents to an SFTP location

GET `http://<baseURL>/v1/documents/destinations/sftp`

- Use the `GET` method to fetch the request body using the following details:
  - URL: `http://<baseURL>/v1/documents/destinations/sftp`

    > **Note**
    >
    > Replace `<baseURL>` with the base URL for RESTful web service requests (such as `host:<port>/biprws`).

  - Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

  - Body: none
    The following response is received from the server:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>
        <title type="text">SendToSftp</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="password" type="string"></attr>
                <attr name="port" type="int32">22</attr>
                <attr name="docid" type="string"></attr>
                <attr name="host" type="string"></attr>
                <attr name="targetobjectname" type="string"></attr>
                <attr name="directory" type="string">.</attr>
                <attr name="fingerPrint" type="string"></attr>
                <attr name="account" type="string"></attr>
                <attr name="username" type="string"></attr>
            </attrs>
        </content>
    </entry>
</feed>
```

POST `http://<baseURL>/v1/documents/destinations/sftp`

- Use the `POST` method to post the send request and pass your specific details in the copied body template.
  Task result: The specific document is sent to the FTP location based on the attribute values passed on the body.

7.14.8 Deleting Documents

DELETE http://<baseURL>/v1/documents/<document_id>

Use the `DELETE` method to delete a document.

Request:

- Method: DELETE
- URL: `DELETE http://<baseURL>/v1/documents/<document_id>`
- Header:

Name	Value
Content Type	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Body:
  An XML entry displays the either a success or failure message for the delete operation. This example shows successful deletion of a document.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="status" type="string">success</attr>
        </attrs>
    </content>
</entry>
```

7.15 Managing Recycle Bin Items

This section provides you information on RESTful APIs that can be used to manage items in the recycle bin.

7.15.1 Listing Recycle Bin Items

GET http://<baseURL>/v1/recyclebin

Request:

- Method: **GET**
- URL: `http://<baseURL>/v1/recyclebin`
  Replace `<baseURL>` with the base URL for RESTful web service requests (such as `10.10.10.10:<port>/biprws`).
- Header:

Name	Value
`Accept`	`application/xml`
`Content-Type`	`application/xml`
`X-SAP-LogonToken`	The logon token value.

- Body: none

Response:

- Header:

Name	Value
`Accept`	`application/xml`
`Content-Type`	`application/xml`
`X-SAP-LogonToken`	The logon token value.

- Body appears displaying the recycle bin items as shown below:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Recycle Bin</title>
    <updated>2016-08-18T10:58:08.349Z</updated>
    <link href="http://10.208.40.31:6405/biprws/v1/recyclebin?
page=1&amp;pagesize=50" rel="self"/>
    <link href="http://10.208.40.31:6405/biprws/v1/recyclebin?
page=1&amp;pagesize=50" rel="first"/>
    <link href="http://10.208.40.31:6405/biprws/v1/recyclebin?
page=1&amp;pagesize=50" rel="last"/>
    <entry>
        <title type="text">1 - 6362</title>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AcIy1BRuniVDgoxERaUq378</attr>
                <attr name="deleteddate" type="string">Thu Aug 18 11:19:49 CEST
2016</attr>
                <attr name="name" type="string">1 - 6362</attr>
                <attr name="id" type="string">6363</attr>
                <attr name="type" type="string">Folder</attr>
                <attr name="ownerid" type="string">12</attr>
                <attr name="parentid" type="string">6359</attr>
```

```
            </attrs>
        </content>
    </entry>
</feed>
```

7.15.2 Retrieving Properties of Recycle Bin Items

GET http://<baseURL>/v1/recyclebin/<doc_ID>

Request:

- Method: **GET**
- URL: `http://<baseURL>/v1/recyclebin/<doc_ID>`
  Replace `<baseURL>` with the base URL for RESTful web service requests (such as `10.10.10.10:<port>/biprws`), and <doc_ID> with the ID of document for which you want to view the properties.
- Header:

Name	Value
`Accept`	`application/xml`
`Content-Type`	`application/xml`
`X-SAP-LogonToken`	The logon token value.

- Body: none

Response:

- The server response displays the item properties such as owner, deleted date, CUID, type, parent etc. An example is shown below:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Recycle Bin</title>
    <updated>2016-08-18T10:58:08.349Z</updated>
    <link href="http://10.208.40.31:6405/biprws/v1/recyclebin?
page=1&amp;pagesize=50" rel="self"/>
    <link href="http://10.208.40.31:6405/biprws/v1/recyclebin?
page=1&amp;pagesize=50" rel="first"/>
    <link href="http://10.208.40.31:6405/biprws/v1/recyclebin?
page=1&amp;pagesize=50" rel="last"/>
    <entry>
        <title type="text">1 - 6362</title>
        <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
              <attr name="owner" type="string">Administrator</attr>
              <attr name="path" type="string">Root Folder/</attr>
              <attr name="cuid" type="string">AcIy1BRuniVDgoxERaUq378</attr>
              <attr name="parentcuid"
type="string">CQFFAAAA9FZpPn7XfU2Ust.ltEwspg</attr>
              <attr name="deleteddate" type="string">Thu Aug 18 11:19:49 CEST
2016</attr>
              <attr name="name" type="string">1 - 6362</attr>
```

```
                <attr name="id" type="string">6363</attr>
                <attr name="ownerid" type="string">12</attr>
                <attr name="type" type="string">Folder</attr>
                <attr name="parentid" type="string">6359</attr>
            </attrs>
            </content>
        </entry>
    </feed>
```

7.15.3 Restoring Items from the Recycle Bin

PUT `http://<baseURL>/v1/recyclebin`

Request:

- Method: **PUT**
- URL: `http://<baseURL>/v1/recyclebin`
  Replace `<baseURL>` with the base URL for RESTful web service requests (such as `10.10.10.10:<port>/biprws`).
- Header:

Name	Value
`Accept`	`application/xml`
`Content-Type`	`application/xml`
`X-SAP-LogonToken`	The logon token value.

- Body: Enter the ID (or CUID) of the items you wish to restore, as shown in the below example:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>

        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="id" type="string">7278</attr>
          </attrs>
        </content>
    </entry>
</feed>
```

Response: A status message appears as shown below:

```
<entry xmlns="http://www.w3.org/2005/Atom">
   <content type="application/xml">
     <attrs xmlns="http://www.sap.com/rws/bip">
     <attr name="result" type="string">success</attr>
     </attrs>
   </content>
</entry
```

7.15.4 Deleting Items from the Recycle Bin

DELETE `http://<baseURL>/v1/recyclebin`

Request:

- Method: **DELETE**
- URL: `http://<baseURL>/v1/recyclebin`
  Replace `<baseURL>` with the base URL for RESTful web service requests (such as `10.10.10.10:<port>/biprws`).
- Header:

Name	Value
`Accept`	`application/xml`
`Content-Type`	`application/xml`
`X-SAP-LogonToken`	The logon token value.

- Body: Enter the ID (or CUID) of the items you wish to delete permanently, as shown in the below example:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>

        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="id" type="string">7278</attr>
        <attr name="read" type="bool">true</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

RESULT: A status message appears in the server response as shown in the below example:

```
<entry xmlns="http://www.w3.org/2005/Atom">
   <content type="application/xml">
     <attrs xmlns="http://www.sap.com/rws/bip">
     <attr name="result" type="string">success</attr>
     </attrs>
   </content>
</entry
```

7.16 Managing Alerts and Notifications

This section provides information on RESTful APIs used to manage alerts and notifications in BI LaunchPad.

7.16.1 Listing Alerts for the Current User

GET http://<base URL>/v1/notification

Request:

- Method: GET
- URL: `GET http://<base URL>/v1/notification`
  Replace `<baseURL>` with the base URL for RESTful web service requests such as `host:port/biprws`.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

RESULT: The available alerts are displayed in the server response:

```
<feed xmlns="http://www.w3.org/2005/Atom">

    <link href="http://10.160.206.203:6405/biprws/v1/notification?
page=1&amp;pagesize=50" rel="self"/>
    <link href="http://10.160.206.203:6405/biprws/v1/notification?
page=1&amp;pagesize=50" rel="first"/>
    <link href="http://10.160.206.203:6405/biprws/v1/notification?
page=1&amp;pagesize=50" rel="last"/>
    <link href="http://10.160.206.203:6405/biprws/v1/notification?
page=1&amp;pagesize=50" rel="self"/>
    <link href="http://10.160.206.203:6405/biprws/v1/notification?
page=1&amp;pagesize=50" rel="first"/>
    <link href="http://10.160.206.203:6405/biprws/v1/notification?
page=1&amp;pagesize=50" rel="last"/>

    <entry>

    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
               <attr name="cuid" type="string">AVQdj2PwwO9GqskQPV3iWKs</attr>

             <attr name="created" type="string">Mon Sep 19 18:16:33 PDT 2016</attr>
                <attr name="name" type="string">MASTERSHAKE.CentralManagementServer
Watch Caution Event</attr>
                <attr name="id" type="string">5957</attr>
                <attr name="message" type="string">%0A%0DThe metrics that have
crossed their respective thresholds: %0D MASTERSHAKE.CentralManagementServer
%24'Current Number of Auditing Events in the Queue'</attr>
                <attr name="updated" type="datetime">2016-09-20T04:56:20.844Z</attr>
        </attrs>
        </content>
    </entry>
    <entry>
        <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
```

```
                <attr name="cuid" type="string">AWJ93gzKmKRBqVwkhgjPLUY</attr>
                <attr name="created" type="string">Mon Sep 19 18:16:38 PDT 2016</
attr>
                <attr name="name" type="string">Core Services Watch Caution Event</
attr>
                <attr name="id" type="string">5958</attr>
                <attr name="message" type="string">%0A%0DThe metrics that have
crossed their respective thresholds: %0DMASTERSHAKE.CentralManagementServer
%24'Health State'</attr>
                <attr name="updated" type="datetime">2016-09-20T04:56:20.860Z</attr>
        </attrs>
        </content>
    </entry>
</feed>
```

7.16.2 Getting Details of an Alert

GET http://<base URL>/v1/notification/<ID>

Use the `GET` method to get the details of a specific alert.

Request:

- Method: GET
- URL: `GET http://host:port/biprws/v1/notification/<id>`
  Replace `<baseURL>` with the base URL for RESTful web service requests such as `host:port/biprws`.

  > **Note**
  >
  > <id> is the ID or CUID of the notification/alert for which you need to view details.

- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

RESULT: The details of the alert are displayed in the received server response as shown in the below example:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="read" type="bool">false</attr>
            <attr name="name" type="string">Consolidated Health Watch Caution
Event</attr>
            <attr name="endtime" type="string"></attr>
            <attr name="id" type="int32">6528</attr>
            <attr name="starttime" type="string"></attr>
```

```
            <attr name="message" type="string">[Reminder  : Caution Rule evaluated
to true for "Consolidated Health Watch" watch., Reminder  :
&#xd;Danger Rule: ServiceCategories$'Health State'==0|| EnterpriseNodes$'Health
State'==0, Reminder  :
&#xd;Caution Rule: ServiceCategories$'Health State'==1|| EnterpriseNodes$'Health
State'==1, Reminder  :
&#xd;The metrics that have crossed their respective thresholds:
&#xd;ServiceCategories$'Health State']</attr>
            <attr name="updated" type="string">Wed Sep 21 21:56:26 PDT 2016</attr>
        </attrs>
    </content>
</entry>
```

7.16.3 Editing Alerts

To mark an alert as read, you can edit it.

POST `http://<base URL>/v1/notification`

Request:

- Method: POST
- URL: `http://<base URL>/v1/notification`
  Replace `<baseURL>` with the base URL for RESTful web service requests such as `host:port/biprws`.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: Pass the ID or CUID of the alerts you need to edit, as shown below:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>

        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="id" type="string">3468</attr>
          </attrs>
        </content>
    </entry>
</feed>
```

> **Tip**
> - To mark a document as "read", pass the value as 'true', and to change a document status to "unread", pass the value as 'false'.
> - To fetch a valid alert ID/CUID, get the alert details as described in the related topic.

RESULT: A status message appears indicating that the alert is successfully edited.

Related Information

Getting Details of an Alert [page 306]

7.16.4 Deleting Alerts

DELETE `http://<base URL>/v1/notification`

Request:

- Method: DELETE
- URL: `http://<baseURL>/v1/notification`
  Replace `<baseURL>` with the base URL for RESTful web service requests such as `host:port/biprws`.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: Pass the **CUID or ID** of the alert you need to delete, as shown below:

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <entry>

        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="id" type="string">7278</attr>
          </attrs>
        </content>
    </entry>
</feed>
```

> **Tip**
>
> To fetch valid alert IDs, get the alert details as described in the related topic.

Response: A status message appears indicating that the alert is successfully deleted.

Related Information

Getting Details of an Alert [page 306]

7.16.5 Getting Notification Banner Details

If you've created a notification banner, it appears at the top of the page when you log on to the BI launch pad as shown below:

GET http://<baseURL>/v1/notification/now

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/notification/now`
  Replace `<baseURL>` with the base URL for RESTful web service requests such as `host:port/biprws`.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

RESULT: The details of the banner are displayed in the server response.

7.16.6 Getting Count of Notification Banners

To get a count of available notification banners, use the below method and URL in your web client application:

GET http://<baseURL>/v1/notification/count

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/notification/count`

Replace `<baseURL>` with the base URL for RESTful web service requests such as `host:port/biprws`.

- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

RESULT: The details of the banner are displayed as shown in the below example:

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="count" type="int32">5</attr>
        </attrs>
    </content>
</entry>
```

7.17 Infostore

7.17.1 Listing Objects In the Infostore

GET http://<baseURL>/v1/infostore

Get a list of objects using the `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/infostore`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Content-Type`	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  An XML feed of all objects created in the BOE system is displayed.


```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">InfoStore (@W2K12R2:6400)</title>
    <updated>2016-05-31T08:51:05.128Z</updated>
    <link href="http://localhost:6405/biprws/v1/infostore?
page=1&amp;pagesize=50" rel="self"/>
    <link href="http://localhost:6405/biprws/v1/infostore?
page=1&amp;pagesize=50" rel="first"/>
    <link href="http://localhost:6405/biprws/v1/infostore?
page=1&amp;pagesize=50" rel="last"/>
    <entry>
        <title type="text">Alert Notifications</title>
        <author>
            <name>System Account</name>
        </author>
        <updated>2016-05-30T08:15:10.953Z</updated>
        <link href="http://localhost:6405/biprws/v1/infostore/Alert
%20Notifications" rel="alternate"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">ARZB.BFCQk9PqaqDpcFwo1w</attr>
                <attr name="name" type="string">Alert Notifications</attr>
                <attr name="description" type="string" null="true"/>
                <attr name="id" type="string">64</attr>
                <attr name="type" type="string">Folder</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">Application Folder</title>
        <author>
            <name>System Account</name>
        </author>
        <updated>2016-05-30T08:15:26.157Z</updated>
        <link href="http://localhost:6405/biprws/v1/infostore/Application
%20Folder" rel="alternate"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">AdoctK9h1sBHp3I6uG0Sh7M</attr>
                <attr name="name" type="string">Application Folder</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">43</attr>
                <attr name="type" type="string">Folder</antr>
            </attrs>
        </content>
    </entry>
</feed>
```


Sorting objetcs

Sorting is the process of arranging the objects in a systematic order.

You can now sort objects sequentially in ascending or descending order based on the following attributes:

- Name
- Description
- ID of the user
- Type

The query parameters for sort and filter are as follows:

Query Parameter

Parameter Name	Parameter Value
Sort	`sort=<+/-><attribute>` **Note** - For ascending order, the value is + - For descending order, the value is - - By default, the objects are sorted in ascending order based on the name of the user group **Example** sort=+name
Paging	- `page=<page number>` - `pagesize=<number>` `pagesize` is the number of objects displayed in the response **Note** By default `page=1` and `pagesize=50` that is, in one page number 1, the top 50 objects will be displayed in the response.

URL: `GET http://<baseURL>/v1/infostore?sort=<name>&page=<page number>&pagesize=<number>`

Example

URL: `GET http://<baseURL>/v1/objects?sort=+name&page=2&pagesize=5`

Once you send the request, the response displays the second page, which has five objects and the list of objects sorted in ascending order.

7.17.2 Getting Object Details

GET http://<baseURL>/v1/infostore/<object_Id>

Get object details using the `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/infostore/<object_Id>`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
Content Type	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  An XML template of object details in the BOE system is displayed. This example shows the object details for the resource with object ID=64.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>System Account</name>
    </author>
    <title type="text">Alert Notifications</title>
    <updated>2016-06-09T04:57:17.054Z</updated>
    <link href="http://localhost:6405/biprws/v1/infostore" rel="up"/>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="cuid" type="string">ARZB.BFCQk9PqaqDpcFwo1w</attr>
            <attr name="name" type="string">Alert Notifications</attr>
            <attr name="description" type="string" null="true"/>
            <attr name="id" type="string">64</attr>
            <attr name="type" type="string">Folder</attr>
```

```
        </attrs>
    </content>
</entry>
```

7.17.3 Listing Children of Objects

GET http://<baseURL>/v1/infostore/<object_Id>/children

Get children details of the object using the `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/infostore/<object_Id>/children`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
Content Type	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  An XML template of the object's children details in BOE system is displayed. This example shows the object details for the resource with object ID=22.

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <title type="text">Children of Calendars</title>
    <updated>2016-06-13T10:18:58.439Z</updated>
    <link href="http://localhost:6405/biprws/v1/infostore/22/children?
page=1&amp;pagesize=50" rel="self"/>
    <link href="http://localhost:6405/biprws/v1/infostore/22/children?
page=1&amp;pagesize=50" rel="first"/>
```

```
    <link href="http://localhost:6405/biprws/v1/infostore/22/children?
page=1&amp;pagesize=50" rel="last"/>
    <entry>
        <title type="text">new calendar</title>
        <author>
            <name>Administrator</name>
            <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
        </author>
        <updated>2016-06-10T09:11:29.229Z</updated>
        <link href="http://localhost:6405/biprws/v1/infostore/6714"
rel="alternate"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="cuid" type="string">ARQOTd0mvLNHlxWFQ7HcgD8</attr>
                <attr name="name" type="string">new calendar</attr>
                <attr name="description" type="string"></attr>
                <attr name="id" type="string">6714</attr>
                <attr name="type" type="string">Calendar</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

Sorting objetcs

Sorting is the process of arranging the object's children in a systematic order.

You can now sort children under a particular object sequentially in ascending or descending order based on the following attributes:

- Name
- Description
- ID of the user
- Type

The query parameters for sort and filter are as follows:

Query Parameter

Parameter Name	Parameter Value
Sort	`sort=<+/-><attribute>` **Note** • For ascending order, the value is + • For descending order, the value is - • By default, the children under a particular object are sorted in ascending based on the name of the user group **Example** sort=+name

Parameter Name	Parameter Value
Paging	• `page=<page number>` • `pagesize=<number>` `pagesize` is the number of children under a particular object displayed in the response **Note** By default, `page=1` and `pagesize=50` that is, in one page number 1, the top 50 childer under a particular object will be displayed in the response.

URL: `GET http://<baseURL>/v1/infostore/<object_Id>/children?sort=<name>&page=<page number>&pagesize=<number>`

Example

URL: `GET http://<baseURL>/v1/22/children?sort=+name&page=2&pagesize=5`

Once you send the request, the response displays the second page, which has five children, and the list of children sorted in ascending order.

7.17.4 Lisitng Relationships of Objects

GET http://<baseURL>/v1/infostore/<object_id>/relationships/<type>

You can retrieve the relationships of a resource by appending /relationships/<type> to the end of the RESTful web service request for the resource.

Get the object details using the `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/infostore/<object_id>/relationships/<type>`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
Content-Type	`application/xml`
`Accept`	`application/xml`

Name	Value
X-SAP-LogonToken	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  An XML template of a object relationship with another object in the BOE system is displayed. This example shows the object details for the resource with object ID=12 and type=userGroups

```
<feed xmlns="http://www.w3.org/2005/Atom">
    <author>
        <name>Administrator</name>
        <uri>http://localhost2:6405/biprws/v1/infostore/12</uri>
    </author>
    <title type="text">InfoObjects related to Administrator via userGroups</
title>
    <updated>2016-06-13T12:02:36.923Z</updated>
    <entry>
        <title type="text">1</title>
        <link href="http://localhost2:6405/biprws/v1/infostore/12/relationships/
userGroups/1" rel="self"/>
        <link href="http://localhost2:6405/biprws/v1/infostore/1" rel="related"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="id" type="string">1</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">2</title>
        <link href="http://localhost2:6405/biprws/v1/infostore/12/relationships/
userGroups/2" rel="self"/>
        <link href="http://localhost2:6405/biprws/v1/infostore/2" rel="related"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="id" type="string">2</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <title type="text">3</title>
        <link href="http://localhost2:6405/biprws/v1/infostore/12/relationships/
userGroups/3" rel="self"/>
        <link href="http://localhost2:6405/biprws/v1/infostore/3" rel="related"/>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="id" type="string">3</attr>
            </attrs>
        </content>
```

```
    </entry>
</feed>
```

7.17.5 Getting Relationship Details Between Objects

GET http://<baseURL>/v1/infostore/<object_id>/relationships/<type>/<another_object_id>

You can retrieve the relationships of a resource by appending /relationships/<type>/<another_object_id> to the end of the RESTful web service request for the resource.

Get object details using the `GET` method.

Request:

- Method: GET
- URL: `GET http://<baseURL>/v1/infostore/<object_id>/relationships/<type>/<another_object_id>`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
Content Type	`application/xml`
`Accept`	`application/xml`
`X-SAP-LogonToken`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Date	Date and time of response.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  An XML template of an object relationship with another object in the BOE system is displayed. This example shows object details for the resource with object ID=12, type=userGroups, and Id=1

```
<entry xmlns="http://www.w3.org/2005/Atom">
```

```
    <title type="text">1</title>
    <author>
        <name>Administrator</name>
        <uri>http://localhost:6405/biprws/v1/infostore/12</uri>
    </author>
    <link href="http://localhost:6405/biprws/v1/infostore/1" rel="related"/>
    <content type="application/xml">
        <attrs xmlns="http://www.sap.com/rws/bip">
            <attr name="id" type="string">1</attr>
        </attrs>
    </content>
</entry>
```

8 CMS Query

This API is used to retreive any objects by supplying a SQL query as the parameter.

To retreive a query result, use `/v1/cmsquery` URI.

- Use the `GET` method to retrieve an XML template for the request body.
- Use the `POST` method to process the query and get the query results.

GET http://<baseURL>/v1/cmsquery

Make a `GET` request to `/v1/cmsquery` to receive a template that can be used in the request body of a `POST` request to the same URL.

Request:

- Method: GET
- URL: `http://<baseURL>/v1/cmsquery`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Accept`	`application/xml`
`Content Type`	`application/xml`
`X-SAP-LOGONTOKEN`	The logon token value, in quotation marks.

- Body: none

Response:

- Header:

Name	Value
Status Code	HTTP response code.
Server	Type of server.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body: An XML template that can be used to populate the request body of the `POST` request.

```
<attrs xmlns="http://www.sap.com/rws/bip">
    <attr name="query" type="string"></attr>
</attrs>
```

POST http://<baseURL>/v1/cmsquery

To receive a query result, make a `POST` request to `/v1/cmsquery`.

Request:

- Method: `POST`
- URL: `http://<baseURL>/v1/cmsquery`
  Replace `<baseURL>` with the base URL for RESTful web service requests.
- Header:

Name	Value
`Accept`	`application/xml`
`Content-Type`	`application/xml`
`X-SAP-LOGONTOKEN`	`LOGONTOKEN`

- Body:

```
<attrs
    xmlns="http://www.sap.com/rws/bip">
    <attr name="query" type="string"><Sample_Query></attr>
</attrs>
```

  - Use `<attr name="query" type="string"><SELECT ATTRIBUTE NAME></attr>` to define the query string.

Response:

- Header:

Attribute	Value
Status Code	HTTP response code.
Server	Type of server.
X-SAP-LogonToken	A logon token.
Content-Type	Type of content in the response body.
Content-Length	Length of content in the response body.

- Body:
  The response body contains the query result.

> **Note**
>
> The sample provided below displays two results for readability purpose, but in the actual scenario, the feed may contain several pages depending on the number of objects returned.

```
<feed  xmlns="http://www.w3.org/2005/Atom">
    <link href="http://localhost:6405/biprws/v1/cmsquery?page=1&pagesize=50"
rel="self" />
    <link href="http://localhost:6405/biprws/v1/cmsquery?page=1&pagesize=50"
rel="first" />
    <link href="http://localhost:6405/biprws/v1/cmsquery?page=2&pagesize=50"
rel="next" />
```

```
    <link href="http://localhost:6405/biprws/v1/cmsquery?page=4&pagesize=50"
rel="last" />
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="SI_DESCRIPTION" type="string"></attr>
                <attr name="SI_ID" type="integer">18</attr>
                <attr name="SI_NAME" type="string">User Folders</attr>
                <attr name="SI_CUID" type="string">AWigQI18AAZJoXfRHLzWJ2c</attr>
                <attr name="SI_SPECIFIC_KIND" type="string">Folder</attr>
                <attr name="SI_OWNERID" type="integer">10</attr>
                <attr name="SI_UPDATE_TS" type="date">Tue May 17 00:02:09 PDT
2016</attr>
            </attrs>
        </content>
    </entry>
    <entry>
        <content type="application/xml">
            <attrs xmlns="http://www.sap.com/rws/bip">
                <attr name="SI_DESCRIPTION" type="string"></attr>
                <attr name="SI_ID" type="integer">23</attr>
                <attr name="SI_NAME" type="string">Root Folder</attr>
                <attr name="SI_CUID" type="string">ASHnC0S_Pw5LhKFbZ.iA_j4</attr>
                <attr name="SI_SPECIFIC_KIND" type="string">Folder</attr>
                <attr name="SI_OWNERID" type="integer">10</attr>
                <attr name="SI_UPDATE_TS" type="date">Tue May 17 00:31:31 PDT
2016</attr>
            </attrs>
        </content>
    </entry>
</feed>
```

9 BI Administrators' Cockpit

9.1 Server

This Sections gives you information on APIs that are used to perform the following operations:

- Server Count [page 323]
- Server List [page 324]
- Server Details [page 329]
- Server Start [page 330]

9.1.1 Server Count

This API is used to list the server count.

To know the server count , use `http://<baseURL>/bionbi/server/` URI.

This API supports GET method.

GET http://<baseURL>/bionbi/server/

If you use this API to make a `GET` request to `/bionbi/server/` URI. You receive the following response:

Response:

- In response, you see the count of total number of servers along with the count of servers in running, stopped, and failed states.
-
```
 <feed>
<id>tag:sap.com,2010:bip-rs/infostore</id>
<title type="text">Metrics for ServersCountByState</title>
<updated>2015-06-11T09:07:01.420Z</updated>
 <entry>
<author>
<name>@win-2k8r2-x64:6400</name>
</author>
<updated>2015-06-11T09:07:01.685Z</updated>
<content type="application/xml">
 <attrs>
<attr name="count" type="int32">19</attr>
<attr name="status_type" type="string">RUNNING</attr>
 </attrs>
</content>
```

```
 </entry>
 <entry>
<author>
<name>@win-2k8r2-x64:6400</name>
</author>
<updated>2015-06-11T09:07:01.685Z</updated>
<content type="application/xml">
 <attrs>
<attr name="count" type="int32">20</attr>
<attr name="status_type" type="string">TOTAL</attr>
 </attrs>
</content>
 </entry>
 <entry>
<author>
<name>@win-2k8r2-x64:6400</name>
</author>
<updated>2015-06-11T09:07:01.685Z</updated>
<content type="application/xml">
 <attrs>
<attr name="count" type="int32">0</attr>
<attr name="status_type" type="string">FAILED</attr>
 </attrs>
</content>
 </entry>
 </feed>
```

9.1.2 Server List

This API is used to list servers.

To list servers, use `http://<baseURL>/bionbi/server/list` URI.

This URI supports GET method.

GET http://<baseURL>/bionbi/server/list

If you use this API to make a `GET` request to `/bionbi/server/list` URI. You receive the following response:

Response:

- In response, you see the list of servers along with its details like: name, type, status, disabled, last modified details, in the CMS.
- **Note**

  In response, you also see the navigation link details, which you can use to navigate between pages.

```
  <feed>
<id>
tag:sap.com,2010:bip-rs/bionbi
</id>
<updated>
2015-07-23T05:00:53.565Z
</updated>
<title type="text">
```

```
Metrics for ServersListForState - ALL
</title>
<link href="http://10.160.192.169:6405/biprws/bionbi/server/list/-1?
page=2&pageSize=5&sortType=DESC" rel="self" />
<link href="http://10.160.192.169:6405/biprws/bionbi/server/list/-1?
page=1&pageSize=5&sortType=DESC" rel="first" />
<link href="http://10.160.192.169:6405/biprws/bionbi/server/list/-1?
page=3&pageSize=5&sortType=DESC" rel="next" />
<link href="http://10.160.192.169:6405/biprws/bionbi/server/list/-1?
page=1&pageSize=5&sortType=DESC" rel="previous" />
<link href="http://10.160.192.169:6405/biprws/bionbi/server/list/-1?
page=4&pageSize=5&sortType=DESC" rel="last" />
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">
AR0rjLlQ8.ZMitZHciycY8E
</attr>
<attr name="parent_id" type="int32">
16
</attr>
<attr name="status_type" type="string">
RUNNING
</attr>
<attr name="description" type="string">
Web Intelligence Processing Server
</attr>
<attr name="disabled" type="string">
Enabled
</attr>
<attr name="id" type="int32">
1242
</attr>
<attr name="title" type="string">
MASTERSHAKE.WebIntelligenceProcessingServer
</attr>
<attr name="last_modified" type="string">
Jul 22, 2015 9:50 PM
</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">
AalIHU17j45Kl7..G_7p6PE
</attr>
<attr name="parent_id" type="int32">
16
</attr>
<attr name="status_type" type="string">
RUNNING
</attr>
<attr name="description" type="string">
Connection Server (32-bit)
</attr>
<attr name="disabled" type="string">
Enabled
</attr>
<attr name="id" type="int32">
1226
</attr>
<attr name="title" type="string">
MASTERSHAKE.ConnectionServer32
</attr>
<attr name="last_modified" type="string">
Jul 22, 2015 9:50 PM
```

```
</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">
Ac9.Y2465eRNvsXAojovONk
</attr>
<attr name="parent_id" type="int32">
16
</attr>
<attr name="status_type" type="string">
RUNNING
</attr>
<attr name="description" type="string">
Connection Server
</attr>
<attr name="disabled" type="string">
Enabled
</attr>
<attr name="id" type="int32">
1205
</attr>
<attr name="title" type="string">
MASTERSHAKE.ConnectionServer
</attr>
<attr name="last_modified" type="string">
Jul 22, 2015 9:50 PM
</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">
Ad3iH88bXNlHnyVOC6r.ibU
</attr>
<attr name="parent_id" type="int32">
16
</attr>
<attr name="status_type" type="string">
RUNNING
</attr>
<attr name="description" type="string">
Event Server
</attr>
<attr name="disabled" type="string">
Enabled
</attr>
<attr name="id" type="int32">
1296
</attr>
<attr name="title" type="string">
MASTERSHAKE.EventServer
</attr>
<attr name="last_modified" type="string">
Jul 22, 2015 9:50 PM
</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">
```

```
AXTecp48TA5Kk17rp9e0ZBo
</attr>
<attr name="parent_id" type="int32">
16
</attr>
<attr name="status_type" type="string">
RUNNING
</attr>
<attr name="description" type="string">
Dashboards Processing Server
</attr>
<attr name="disabled" type="string">
Enabled
</attr>
<attr name="id" type="int32">
1169
</attr>
<attr name="title" type="string">
MASTERSHAKE.DashboardsProcessingServer
</attr>
<attr name="last_modified" type="string">
Jul 22, 2015 9:50 PM
</attr>
 </attrs>
</content>
</entry>
 </feed>
```


Using this URI, you can perform various operations like:

- Listing servers based on their state
- Listing servers page by page
- Sorting servers

Listing Servers Based on Their State:

Using this API, you can list servers of particular state.

The following are the predefined server state values:

Server State Values table

Keyword	Predefined Values
Public static final int STOPPED	0
Public static final int STARTING	1
Public static final int INITIALIZING	2
Public static final int RUNNING	3
Public static final int STOPPING	4
Public static final int FAILED	5
Public static final int RUNNING_WITH_ERRORS	6
Public static final int RUNNING_WITH_WARNINGS	7

Use `http://10.160.212.35:6405/biprws/bionbi/server/list/<Server_State_value>`, to list servers of particular state.

Parameter Description

Parameter Name	Description
`<Server_State_value>`	Server state value(see the *Server State Value* table).

For example: Use `http://<baseURL>/bionbi/server/list/3` to list servers in RUNNING state.

For more information on the server state, refer the *Server State Values table*.

Listing Servers Page by Page:

You can either use the navigation link obtained in the response or you can manual enter `page` and `pageSize` parameter details as shown below:

To list servers page by page use the following URI:

`http://<baseURL>/bionbi/server/list?page=<Page_no>&pageSize=<Max_Page_Limit>`

Following table gives you the parameters description:

Parameter Description

Parameter Name	Value	Description
`page`	`<Page_no>`	The parameter `page` specifies the page number.
`pageSize`	`<Max_Page_Limit>`	The parameter `pageSize` specifies the maximum limit of a page. **Note** By default, `pageSize=50`

For example 1: Use `http://<baseURL>/bionbi/server/list?page=1&pageSize=10`, lists first 10 server details in one page.

For example 2: Use `http://<baseURL>/bionbi/server/list?page=2&pageSize=10`, lists 11 through 20 servers in second page.

Sorting Servers:

Sorting is based on the server's modified time.

Using `http://<baseURL>/bionbi/server/list&sortType=<Sort_Type_value>` URI, you can perform sorting in ascending or descending order.

Following table gives you the parameters description:

Parameter Description

Parameter Name	Value	Description
sortType	`<Sort_Type_value>`	This parameter either takes the value `ASC`(Ascending) or `DESC`(Descending) as its value. **Note** By default `<sortType>` is set to DESC.

For example:`http://<baseURL>/bionbi/server/list&sortType=DESC`, lists servers in descending order, based on their modified time.

Related Information

Server Count [page 323]

9.1.3 Server Details

To list details of a server like: Status , Error Message, title, description, ID, Last Modified, Type, use `http://<baseURL>/bionbi/server/<Server_ID>` URI.

This API supports GET method.

GET :http://<baseURL>/bionbi/server/<Server_ID>

Parameter Description

Parameter Name	Description
<Server_ID>	SI_ID of a server.

For example: If you want to know the details of a server with ID=1162, use the ID of the server in the URI as shown below:

`http://10.160.212.35:6405/biprws/bionbi/server/1162`. Use `GET` method to send the request.

Response: In response, you receive the following details:

- ```
  <feed>
  <id>tag:sap.com,2010:bip-rs/infostore</id>
  <title type="text">Metrics for ServerDetailsForServer - ID:1162</title>
  <updated>2015-06-11T09:36:55.869Z</updated>
  <entry>
  <content type="application/xml">
  ```
  ```

```
 <attrs>
<attr name="id" type="int32">1162</attr>
<attr name="last_modified" type="datetime">2015-06-11T06:45:17.256Z</attr>
<attr name="cuid" type="string">AaY29jP964ZKsOM..uVmCjg</attr>
<attr name="title" type="string">WIN2K8R2X64.AdaptiveProcessingServer</attr>
<attr name="status_type" type="string">RUNNING</attr>
<attr name="description" type="string">Adaptive Processing Server</attr>
<attr name="disabled" type="bool">false</attr>
<attr name="parent_id" type="int32">16</attr>
 </attrs>
</content>
</entry>
 </feed>
```

Related Information

Server Count [page 323]
Server List [page 324]

9.1.4 Server Start

To start a server, which is in failed or stopped state, use`http://<baseURL>/bionbi/server/start/<Server_ID>` URI.

This API supports `GET` method.

GET: http://<baseURL>/bionbi/server/start/<Server_ID>

Parameter Description

Parameter Name	Description
<Server_ID>	SI_ID of a server.

For example: If you want to start a server, with ID=1180, use the ID of the server in the URI as shown below:

`http://10.160.212.35:6405/biprws/bionbi/server/start/1180`. Use `GET` method to send the request.

Response: In response, you receive the following details:

- ```
  <feed>
  <id>tag:sap.com,2010:bip-rs/infostore</id>
  <title type="text">Metrics for StartServer - ID:1180</title>
  <updated>2015-06-11T10:33:11.485Z</updated>
  <entry>
  <content type="application/xml">
   <attrs>
  <attr name="id" type="int32">1180</attr>
  <attr name="title" type="string">WIN2K8R2X64.EventServer</attr>
  <attr name="action_status" type="string">Success</attr>
  ```
  ```

```
 </attrs>
</content>
</entry>
 </feed>
```

> **Note**
>
> If the `<action_status>` is success, this means the start action was triggered successfully. To view the actual status of the server, use Server Details or Server List URI.

Related Information

Server Count [page 323]
Server List [page 324]
Server Details [page 329]

9.2 Scheduled Jobs

9.2.1 Job Count

This API is used to display the count of the jobs that was scheduled for the given period.

> **Note**
>
> The provided date and time refers to UTC time zone. By default, period range is as follows:
>
> - `<startDate>` is 1/1/1900 00:00:00
> - `<endDate>`is 1/1/2900 00:00:00

To know the job count, use `http://<baseURL>/bionbi/job/` URI.

GET http://<baseURL>/bionbi/job/

If you use this API to make a `GET` request to `/bionbi/job/` URI. You receive the following response:

Response:

- In response, you see the count of total number of jobs along with the count for jobs in success, running, and failed states.

-
```
<feed>
<id>tag:sap.com,2010:bip-rs/bionbi</id>
<title type="text">Metrics for JobsCountByState</title>
<updated>2015-06-24T08:25:04.652Z</updated>
<author>
<name>@win-2k12r2-x64:6400</name>
</author>
<entry>
<content type="application/xml">
 <attrs>
<attr name="count" type="int32">1</attr>
<attr name="status_type" type="string">SUCCESS</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="count" type="int32">0</attr>
<attr name="status_type" type="string">RUNNING</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="count" type="int32">9</attr>
<attr name="status_type" type="string">TOTAL</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="count" type="int32">8</attr>
<attr name="status_type" type="string">FAILED</attr>
 </attrs>
</content>
</entry>
 </feed>
```


If you want to know the count of jobs scheduled at particular duration, then use the query parameters `<startDate><endDate>`in the URI as follows:

> **Note**
>
> The provided date and time refers to UTC time zone.

```
http://<baseURL>/bionbi/job?startDate=<MM/DD/YYYY HH:MM:SS>&endDate=<MM/DD/YYYY
HH:MM:SS>
```

Parameter Description

Parameter Name	Value	Description
startDate	`<MM/DD/YYYY HH:MM:SS>`	This parameter is used to specify the start date in MM/DD/YYYY HH:MM:SS format.
endDate	`<MM/DD/YYYY HH:MM:SS>`	This parameter is used to specify the end date in MM/DD/YYYY HH:MM:SS format.

For example: `http://<baseURL>/bionbi/job?startDate=06/12/2015 00:00:00&endDate=07/15/2015 00:00:00` , counts all jobs scheduled between June 12th, 2015 and July 15th, 2015 for UTC time zone.

> **Note**
>
> A job is considerd only if its last scheduled time lies in between the specified duration.

9.2.2 Job List

This API is used to list available jobs that was scheduled for the given period.

To list available jobs, use `http://<baseURL>/bionbi/job/list` URI.

GET http://<baseURL>/bionbi/job/list

If you use this API to make a `GET` request to `/bionbi/job/list` URI. You receive the following response:

Response:

- In response, you see the list of jobs along with its details like: status, name, type, owner, creation time details.
- > **Note**
  >
  > In response, you also see the navigation link details, which you can use to navigate between pages.

```
 <feed>
<id>tag:sap.com,2010:bip-rs/bionbi</id>
<updated>2015-09-21T09:34:17.991Z</updated>
<title type="text">Metrics for JobsListForState - ALL</title>
<link href="http://10.160.209.144:6405/biprws/bionbi/job/list/-1?
page=1&pageSize=50&filterVals&startDate=07/14/2015+15%3A00%3A00&endDate=09/15/201
5+23%3A00%3A00&sortType=DESC" rel="self" />
<link href="http://10.160.209.144:6405/biprws/bionbi/job/list/-1?
page=1&pageSize=50&filterVals&startDate=07/14/2015+15%3A00%3A00&endDate=09/15/201
5+23%3A00%3A00&sortType=DESC" rel="first" />
<link href="http://10.160.209.144:6405/biprws/bionbi/job/list/-1?
page=2&pageSize=50&filterVals&startDate=07/14/2015+15%3A00%3A00&endDate=09/15/201
5+23%3A00%3A00&sortType=DESC" rel="next" />
<link href="http://10.160.209.144:6405/biprws/bionbi/job/list/-1?
page=313&pageSize=50&filterVals&startDate=07/14/2015+15%3A00%3A00&endDate=09/15/2
015+23%3A00%3A00&sortType=DESC" rel="last" />
<entry>
<content type="application/xml">
 <attrs>
<attr name="creation_time" type="string">Sep 15, 2015 12:26 PM</attr>
<attr name="owner" type="string">Administrator</attr>
<attr name="cuid" type="string">AQp1wLrvEFNKrX_3gOVhJ68</attr>
<attr name="parent_id" type="int32">17412</attr>
<attr name="status_type" type="string">Success</attr>
<attr name="end_time" type="string">Sep 15, 2015 12:28 PM</attr>
<attr name="description" type="string" />
<attr name="id" type="int32">345670</attr>
<attr name="title" type="string">Pub_webi [1]</attr>
<attr name="type" type="string">Publication</attr>
```

```
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="creation_time" type="string">Sep 8, 2015 5:41 AM</attr>
<attr name="owner" type="string">Administrator</attr>
<attr name="cuid" type="string">ATewtgSXKilOle6Q6ngiPkQ</attr>
<attr name="parent_id" type="int32">342842</attr>
<attr name="status_type" type="string">Failed</attr>
<attr name="end_time" type="string">Sep 8, 2015 5:41 AM</attr>
<attr name="description" type="string" />
<attr name="id" type="int32">342909</attr>
<attr name="title" type="string">pub2</attr>
<attr name="type" type="string">Publication</attr>
 </attrs>
</content>
</entry>
</feed>
```


Using this URI, you can perform various operations like:

- Listing jobs page by page
- Filtering jobs based on report kind
- Listing jobs sheduled at particular duartion

Listing Jobs Page by Page:

You can either use the navigation link obtained in the response or you can manual enter page and pageSize parameter details as shown below:

To list jobs page by page use the following URI:

```
http://10.160.212.35:6405/biprws/bionbi/job/list?
page=<Page_no>&pageSize=<Max_Page_Limit>
```

Following table gives you the description for the parameters `page` and `pageSize`:

Parameter Description

Parameter Name	Value	Description
`page`	`<Page_no>`	The parameter `page` specifies the page number.
`pageSize`	`<Max_Page_Limit>`	The parameter `pageSize` specifies the maximum limit of a page. **Note** By default, `pageSize=50`

For example 1: Use `http://10.160.212.35:6405/biprws/bionbi/job/list?page=1&pageSize=10`, lists first 10 jobs details in one page.

For example 2: Use `http://10.160.212.35:6405/biprws/bionbi/job/list?page=2&pageSize=10`, lists 11 through 20 jobs in second page.

Filtering Jobs Based on Report Kind:

You can filter the jobs based on the particular kind of the job. **For example:** If you want to view jobs of kind `<CrystalReport >`only, then specify the value for query parameter as,`<filterVals>`=CrystalReport.

> **Note**
>
> - By default, it displays all kinds of reports.
> - You can specify one or more kind in the `<filterVals>` query parameter, but these must be comma separated (,).

Report Kind

Report Kind
Webi
CrystalReport
Publication
AAD.AnalysisApplication
Xcelsius
XL.XcelsiusExtension
XL.XcelsiusEnterprise
DataDiscovery
DataDiscoveryAlbum
AO.Workbook
MDAnalysis
VISILums
LumsExtension
Hyperlink
Note
pQuery
AO.Presentation
FullClient

To list jobs of particular kinds, use `http://10.160.212.35:6405/biprws/bionbi/job/list?filterVals=<Report_Kind1,Report_Kind2,Report_Kind3,>` URI.

Parameter Description

Parameter Name	Value	Description
filterVals	<Report_Kind1,Report_Kind2,Report_Kind3,>	Kinds of reports. Refer to **Report Kind** table.

For example: To list jobs of report kind crystal report, use `http://10.160.212.35:6405/biprws/bionbi/job/list&filterVals=Crystalreport`in the URI .

Listing Jobs Sheduled at Particular Duartion:

If you want to know the list of jobs scheduled at particular duration, then use the query parameters `<startDate><endDate>`in the URI as follows:

> **Note**
>
> The provided date and time refers to UTC time zone.

```
http://<baseURL>/bionbi/job?startDate=<MM/DD/YYYY HH:MM:SS>&endDate=<MM/DD/YYYY HH:MM:SS>
```

Parameter Description

Parameter Name	Value	Description
startDate	<MM/DD/YYYY HH:MM:SS>	This parameter is used to specify the start date in MM/DD/YYYY HH:MM:SS format.
endDate	<MM/DD/YYYY HH:MM:SS>	This parameter is used to specify the end date in MM/DD/YYYY HH:MM:SS format.

For example: `http://<baseURL>/bionbi/job/list?startDate=06/12/2015 00:00:00&endDate=07/15/2015 00:00:00` , counts all jobs scheduled between June 12th, 2015 and July 15th, 2015 for UTC time zone.

> **Note**
>
> - A job is considerd only if its last scheduled time lies in between the specified duration.
> - By default, `<startDate>` is 1/1/1900 00:00:00
> - By default,`<endDate>`is 1/1/2900 00:00:00

9.2.3 Job Details

To list details of a job like: status, error message, title, description, ID, creation time, type, use `http://<baseURL>/bionbi/job/<Job_ID>` URI.

GET :http://<baseURL>/bionbi/job/<Job_ID>

Parameter Description

Parameter Name	Description
<Job_ID>	Job ID.

For example: If you want to know the details of a job with ID=6533, use the job ID in the URI as shown below:

`http://10.160.212.35:6405/biprws/bionbi/job/6533`. Use `GET` method to send the request.

Response: In response, you receive the following details:

- ```
  <feed>
  <id>tag:sap.com,2010:bip-rs/bionbi</id>
  <title type="text">Metrics for JobDetails - ID:6533</title>
  <updated>2015-06-25T02:24:41.916Z</updated>
  <entry>
  <content type="application/xml">
  <attrs>
  <attr name="id" type="int32">6533</attr>
  <attr name="cuid" type="string">AZKbv6FL0dJDuywNom.2XN0</attr>
  <attr name="title" type="string">Publication_Test</attr>
  <attr name="end_time" type="datetime">2025-04-09T05:37:29.347Z</attr>
  <attr name="status_type" type="string">RUNNING</attr>
  <attr name="description" type="string" />
  <attr name="owner" type="string">Administrator</attr>
  <attr name="type" type="string">Publication</attr>
  <attr name="error_msg" type="string" />
  <attr name="parent_id" type="int32">6524</attr>
  </attrs>
  </content>
  </entry>
  </feed>
  ```

9.2.4 Job Re-Run

To re-run the job, use `http://10.160.212.35:6405/biprws/bionbi/job/rerun/<job_ID>` URI.

GET http://<baseURL>/biprws/bionbi/job/rerun/<job_ID>

Use this API to make a `GET` request to `/bionbi/job/rerun/<job_ID>` URI.

Parameter Description

Parameter Name	Description
Job_ID	Job ID

For example: To re-run the the job with job ID=6533, use `http://10.160.212.35:6405/biprws/bionbi/job/rerun/6533` URI.
  ```

Response: In response, you receive the following details:

- ```
  <feed>
  <id>tag:sap.com,2010:bip-rs/bionbi</id>
  <title type="text">Metrics for RerunJob - ID:6533</title>
  <updated>2015-06-24T09:22:52.088Z</updated>
  <entry>
  <content type="application/xml">
   <attrs>
  <attr name="id" type="int32">6533</attr>
  <attr name="title" type="string">Publication_Test</attr>
  <attr name="action_status" type="string">Success</attr>
   </attrs>
  </content>
  </entry>
   </feed>
  ```

9.2.5 Statistics Highest Instances

To list the jobs having maximum number of instances, with highest instance listed on top, use `http://10.160.212.35:6405/biprws/bionbi/job/stats/highestinstances` URI.

GET http://<baseURL>/biprws/bionbi/job/stats/highestinstances

If you use this API to make a `GET` request to `/bionbi/job/stats/highestinstances` URI. You receive the following response:

Response:

- In response, you see the list of jobs with maximum number of instances, with highest instance listed on top along with its details like: id, cuid, title, description, report_count, type, parent_id details.
- ```
  <feed>
  <id>tag:sap.com,2010:bip-rs/bionbi</id>
  <updated>2015-06-24T09:17:14.580Z</updated>
  <title type="text">Metrics for Job Statistics - Reports with Highest Instances</
  title>
  <entry>
  <content type="application/xml">
   <attrs>
  <attr name="id" type="int32">6533</attr>
  <attr name="cuid" type="string">AZKbv6FL0dJDuywNom.2XN0</attr>
  <attr name="title" type="string">Publication_Test</attr>
  <attr name="description" type="string" />
  <attr name="report_count" type="int32">3</attr>
  <attr name="type" type="string">Publication</attr>
  <attr name="parent_id" type="int32">6524</attr>
   </attrs>
  </content>
  </entry>
  <entry>
  <content type="application/xml">
   <attrs>
  <attr name="id" type="int32">6785</attr>
  <attr name="cuid" type="string">AbmTm6vpi9NKiWvK76t3aYc</attr>
  <attr name="title" type="string">Publication test</attr>
  ```

```
<attr name="description" type="string" />
<attr name="report_count" type="int32">2</attr>
<attr name="type" type="string">Publication</attr>
<attr name="parent_id" type="int32">6786</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="id" type="int32">7216</attr>
<attr name="cuid" type="string">AQMVpjlCv_RAuytB9eJ96Eo</attr>
<attr name="title" type="string">Publication test_new</attr>
<attr name="description" type="string" />
<attr name="report_count" type="int32">1</attr>
<attr name="type" type="string">Publication</attr>
<attr name="parent_id" type="int32">6786</attr>
 </attrs>
</content>
</entry>
 </feed>
```

Using this URI, you can perform various operations like:

- Listing top `N` jobs that have maximum number of instances
- Listing job sheduled at particular duartion

Listing Top `N` Jobs That Have Maximum Number of Instances:

If you want to list top N jobs that have maximum number of instances, use `http://10.160.212.35:6405/biprws/bionbi/job/stats/highestinstances?topN=<Top_Value>`

> **Note**
>
> By default all jobs are listed.

Parameter Description

Parameter Name	Value	Description
topN	`<Top_Value>`	Lists top `1 to N` number of jobs that have maximum number of instances.

Listing Job's Instances Sheduled at Particular Duration:

If you want to list job's instances scheduled at particular duration, then use the query parameters `<startDate><endDate>`in the URI as follows:

> **Note**
>
> The provided date and time refers to UTC time zone.

```
http://<baseURL>/biprws/bionbi/job/stats/highestinstances?startDate=<MM/DD/YYYY
HH:MM:SS>&endDate=<MM/DD/YYYY HH:MM:SS>
```

Parameter Description

Parameter Name	Value	Description
startDate	`<MM/DD/YYYY HH:MM:SS>`	This parameter is used to specify the start date in MM/DD/YYYY HH:MM:SS format.
end Date	`<MM/DD/YYYY HH:MM:SS>`	This parameter is used to specify the end date in MM/DD/YYYY HH:MM:SS format.

For example: `http://<baseURL>/biprws/bionbi/job/stats/highestinstances?startDate=06/12/2015 00:00:00&endDate=07/15/2015 00:00:00`, counts all jobs scheduled between June 12th, 2015 and July 15th, 2015 having highest instances for UTC time zone.

> **Note**
>
> - A job is considerd only if its last scheduled time lies in between the specified duration.
> - By default, `<startDate>` is 1/1/1900 00:00:00
> - By default,`<endDate>`is 1/1/2900 00:00:00

9.2.6 Statistics Longest Instances

To list jobs having longest instances, with longest running instance listed on top, use `http://10.160.212.35:6405/biprws/bionbi/job/stats/longestinstances` URI.

GET http://<baseURL>/bionbi/job/stats/longestinstance

If you use this API to make a `GET` request to `/bionbi/job/stats/longestinstances` URI. You receive the following response:

Response:

- In response, you see the list of jobs having longest instances, with longest running instance listed on top along with its details like: id, cuid, title, description, schedule_duration, type, parent_id details.
-
```
 <feed>
<id>tag:sap.com,2010:bip-rs/bionbi</id>
<updated>2015-06-24T09:13:12.438Z</updated>
<title type="text">Metrics for Job Statistics - Longest running Instances</title>
<entry>
<content type="application/xml">
 <attrs>
<attr name="id" type="int32">6872</attr>
<attr name="cuid" type="string">AaXISFS8GrpMi6UwMeOcPpI</attr>
<attr name="title" type="string">Publication_Test</attr>
<attr name="description" type="string" />
<attr name="schedule_duration" type="int32">2086587392</attr>
<attr name="type" type="string">Publication</attr>
<attr name="parent_id" type="int32">6533</attr>
 </attrs>
```

```
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="id" type="int32">6884</attr>
<attr name="cuid" type="string">AcKekb2zb0JLgLj0eA0fqJI</attr>
<attr name="title" type="string">Comparative Income Statement[0]</attr>
<attr name="description" type="string" />
<attr name="schedule_duration" type="int32">2086587392</attr>
<attr name="type" type="string">CrystalReport</attr>
<attr name="parent_id" type="int32">6883</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="id" type="int32">6885</attr>
<attr name="cuid" type="string">AXc3VywLD0hGoNFoFV6rqI0</attr>
<attr name="title" type="string">Comparative Income Statement[1]</attr>
<attr name="description" type="string" />
<attr name="schedule_duration" type="int32">2086587392</attr>
<attr name="type" type="string">CrystalReport</attr>
<attr name="parent_id" type="int32">6883</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="id" type="int32">6886</attr>
<attr name="cuid" type="string">Ab.C5uLcjg5GolpkF1H5YVs</attr>
<attr name="title" type="string">Comparative Income Statement[2]</attr>
<attr name="description" type="string" />
<attr name="schedule_duration" type="int32">2086587392</attr>
<attr name="type" type="string">CrystalReport</attr>
<attr name="parent_id" type="int32">6883</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="id" type="int32">6887</attr>
<attr name="cuid" type="string">AcLBhXBErOpBhRLl10OBhSw</attr>
<attr name="title" type="string">Comparative Income Statement[3]</attr>
<attr name="description" type="string" />
<attr name="schedule_duration" type="int32">2086587392</attr>
<attr name="type" type="string">CrystalReport</antr>
<attr name="parent_id" type="int32">6883</attr>
 </attrs>
</content>
</entry>
 </feed>
```


Using this URI, you can perform various operations like:

- Listing top N jobs that have instances running for longer duration
- Listing jobs sheduled at particular duartion

Listing Top N Jobs That Have Instances Running for Longer Duration:

If you want to list top N jobs that are running for a longer duration, use `http://10.160.212.35:6405/biprws/bionbi/job/stats/longestinstances?topN=<Top_Value>`

> **Note**
>
> By default all jobs are listed.

Parameter Description

Parameter Name	Value	Description
topN	`<Top_Value>`	Lists top `1 to N` number of jobs that have maximum number of instances.

Listing Jobs Sheduled at Particular Duration:

If you want to list jobs running for a longer duration at particular time, then use the query parameters `<startDate><endDate>`in the URI as follows:

> **Note**
>
> The provided date and time refers to UTC time zone.

```
http://<baseURL>/job/stats/longestinstances?startDate=<MM/DD/YYYY
HH:MM:SS>&endDate=<MM/DD/YYYY HH:MM:SS>
```

Parameter Description

Parameter Name	Value	Description
startDate	`<MM/DD/YYYY HH:MM:SS>`	This parameter is used to specify the start date in MM/DD/YYYY HH:MM:SS format.
end Date	`<MM/DD/YYYY HH:MM:SS>`	This parameter is used to specify the end date in MM/DD/YYYY HH:MM:SS format.

For example: `http://<baseURL>/bionbi/job/stats/longestinstances?startDate=06/12/2015 00:00:00&endDate=07/15/2015 00:00:00` , counts all jobs scheduled between June 12th, 2015 and July 15th, 2015 and having longest running time for UTC time zone.

> **Note**
>
> - A job is considerd only if its last scheduled time lies in between the specified duration.
> - By default, `<startDate>` is 1/1/1900 00:00:00
> - By default,`<endDate>`is 1/1/2900 00:00:00

9.3 Content Usage

9.3.1 Content Count

This API is used to obtain the count active and inactive report documents that are viewed, refreshed or scheduled.

To obtain the report document count, use `http://<baseURL>/biprws/bionbi/content` URI.

GET http://<baseURL>/biprws/bionbi/content

If you use this API to make a `GET` request to `/bionbi/content` URI. You receive the following response:

Response:

- In response, you see the count of active and inactive report documents that are viewed, refreshed or scheduled.

  > **Note**
  >
  > The reports are in active state, if the report documents are viewed, refreshed or scheduled within a given period of time, otherwise the report documents are said to be in an inactive state.

- ```
  <feed>
  <id>
  tag:Sap.com,2010:bip-rs/bionbi
  </id>
  <title type="text">
  Metrics for ContentCountByState
  </title>
  <updated>
  2015-08-19T05:55:17.716Z
  </updated>
  <author>
  <name>
  @win-2k8r2-x64:6400
  </name>
  </author>
  <entry>
  <content type="application/xml">
   <attrs>
  <attr name="status_type" type="string">
  ACTIVE
  </attr>
  <attr name="count" type="int32">
  5
  </attr>
   </attrs>
  </content>
  </entry>
  <entry>
  <content type="application/xml">
   <attrs>
  <attr name="status_type" type="string">
  INACTIVE
  </attr>
  <attr name="count" type="int32">
  36
  </attr>
   </attrs>
  </content>
  </entry>
  ```
  ```

```
</feed>
```

Using this URI, you can perform the following operation:

If you want to know the active and inactive report document count that are viewed, refreshed, or scheduled at particular duration, then use the query parameters `<startDate><endDate>`in the URI as follows:

> **Note**
>
> The provided date and time refers to UTC time zone.

```
http://<baseURL>/biprws/bionbi/content&startDate=<MM/DD/YYYY HH:MM:SS>&endDate=<MM/DD/YYYY HH:MM:SS>
```

Parameter Description

Parameter Name	Value	Description
startDate	`<MM/DD/YYYY HH:MM:SS>`	This parameter is used to specify the start date in MM/DD/YYYY HH:MM:SS format.
end Date	`<MM/DD/YYYY HH:MM:SS>`	This parameter is used to specify the end date in MM/DD/YYYY HH:MM:SS format.

For example: `http://<baseURL>/biprws/bionbi/content&startDate=06/12/2015 00:00:00&endDate=07/15/2015 00:00:00` , displays the report document count that are viewed, refreshed, or scheduled between June 12th, 2015 and July 15th, 2015(active state) for UTC time zone, otherwise the report documents are considered to be inactive.

> **Note**
>
> - By default, `<startDate>` is 1/1/1900 00:00:00
> - By default,`<endDate>`is 1/1/2900 00:00:00

9.3.2 Content List

To list active or inactive InfoObjects, use `http://<baseURL>/bionbi/content/list` URI.

> **Note**
>
> By default, this API lists inactive InfoObjects.

GET http://<baseURL>/bionbi/content/list

If you use this API to make a `GET` request to `/bionbi/content/list` URI. You receive the following response:

Response:

- **Note**

  In response, you also see the navigation link details, which you can use to navigate between pages.

  This API returns the following types of report kinds:

  Report Kind

Report Kind
Webi
CrystalReport
AAD.AnalysisApplication
XL.XcelsiusEnterprise
DataDiscovery
Xcelsius
DataDiscoveryAlbum
MDAnalysis
AO.Workbook
VISILums
LumsExtension
Hyperlink
Note
pQuery
AO.Presentation
FullClient

```
<feed>
<id>tag:sap.com,2010:bip-rs/bionbi</id>
<updated>2015-08-19T06:02:22.635Z</updated>
<title type="text">Metrics for ContentListForState - ACTIVE</title>
<link href="http://10.160.196.46:6405/biprws/bionbi/content/list/511?
page=1&pageSize=50&topN=0&startDate=08/18/2015&endDate=08/18/2015&sortType=DESC"
rel="self" />
<link href="http://10.160.196.46:6405/biprws/bionbi/content/list/511?
page=1&pageSize=50&topN=0&startDate=08/18/2015&endDate=08/18/2015&sortType=DESC"
rel="first" />
<link href="http://10.160.196.46:6405/biprws/bionbi/content/list/511?
page=1&pageSize=50&topN=0&startDate=08/18/2015&endDate=08/18/2015&sortType=DESC"
rel="last" />
```

```
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">AQtkbbSqN4NOj3ydf.Sw1lY</attr>
<attr name="status_type" type="string">Active</attr>
<attr name="last_run" type="string">Aug 18, 2015 12:00 AM</attr>
<attr name="title" type="string">Formatting Sample</attr>
<attr name="type" type="string">Web Intelligence</attr>
<attr name="view_count" type="int32">4</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">AYxI.vxl9IRCq0sSBP9Mybs</attr>
<attr name="status_type" type="string">Active</attr>
<attr name="last_run" type="string">Aug 18, 2015 12:00 AM</attr>
<attr name="title" type="string">Charting Samples</attr>
<attr name="type" type="string">Web Intelligence</attr>
<attr name="view_count" type="int32">2</attr>
 </attrs>
</content>
</entry>
<entry>
</entry>
 </feed>
```


Using this URI, you can perform various operations like:

- Listing active InfoObjects
- Listing top N active InfoObjects
- Sorting InfoObjects
- Listing InfoObjects page by page
- Listing InfoObjects used at particular duartion

Listing Active InfoObjects

To list the active InfoObjects, use the following URI:

`http://<baseURL>/bionbi/content/list/1`

> **Note**
>
> By default, the API lists inactive InfoObjects.
>
> or you can use `http://<baseURL>/bionbi/content/list/2` to list inactive documents.

Listing top N Active InfoObjects

You can list top N active InfoObjects by using the following URI:

`http://10.160.212.35:6405/biprws/bionbi/content/list?topN=<Top_Value>`

Sorting InfoObjects

Sorting for active InfoObjects is based on view count and sorting for inactive InfoObjects is based on last runtime.

Using `http://<baseURL>/bionbi/content/list?sortType=<Sort_Type_value>` URI, you can perform sorting in ascending or descending order.

Following table gives you the parameters description:

Parameter Description

Parameter Name	Value	Description
sortType	`<Sort_Type_value>`	This parameter either takes the value `ASC`(Ascending) or `DESC`(Descending) as its value. **Note** By default `<sortType>` is set to DESC.

For example:`http://<baseURL>/biprws/bionbi/content/list?sortType=DESC`, lists inactive InfoObjects in descending order, based on their last runtime.

For example:`http://<baseURL>/biprws/bionbi/content/list/1?sortType=DESC`, lists all active InfoObjects in descending order, based on their view count.

Listing InfoObjects Page by Page:

You can either use the navigation link obtained in the response or you can manual enter `page` and `pageSize` parameter details as shown below:

To list InfoObjects page by page use the following URI:

```
http://<baseURL>/biprws/bionbi/content/list?page=<Page_no>&pageSize=<Max_Page_Limit>
```

Following table gives you the description for the parameters `page` and `pageSize`:

Parameter Description

Parameter Name	Value	Description
`page`	`<Page_no>`	The parameter `page` specifies the page number.
`pageSize`	`<Max_Page_Limit>`	The parameter `pageSize` specifies the maximum limit of a page. **Note** By default, `pageSize=50`

For example 1: Use `http://<baseURL>/biprws/bionbi/content/list?page=1&pageSize=10`, lists first 10 inactive InfoObjects in one page.

For example 2: Use `http://<baseURL>/biprws/bionbi/content/list?page=2&pageSize=10`, lists 11 through 20 inactive InfoObjects in the second page.

Listing InfoObjects at Particular Duration:

If you want to know the list of InfoObjects accessed at particular duration, then use the query parameters `<startDate><endDate>`in the URI as follows:

> **Note**
>
> The provided date and time refers to UTC time zone.

```
http://<baseURL>/biprws/bionbi/content/list?startDate=<MM/DD/YYYY
HH:MM:SS>&endDate=<MM/DD/YYYY HH:MM:SS>
```

Parameter Description

Parameter Name	Value	Description
startDate	`<MM/DD/YYYY HH:MM:SS>`	This parameter is used to specify the start date in MM/DD/YYYY HH:MM:SS format.
end Date	`<MM/DD/YYYY HH:MM:SS>`	This parameter is used to specify the end date in MM/DD/YYYY HH:MM:SS format.

For example: `http://<baseURL>/biprws/bionbi/content/list?startDate=06/12/2015 00:00:00&endDate=07/15/2015 00:00:00` , lists all inactive InfoObjects accessed between June 12th, 2015 and July 15th, 2015 for UTC time zone.

> **Note**
>
> - By default, `<startDate>` is 1/1/1900 00:00:00
> - By default,`<endDate>`is 1/1/2900 00:00:00

9.3.3 Statistics Folders by Reports

To list the folders having maximum number of report documents, with highest count of report documents listed on top, use `http://<baseURL>/biprws/bionbi/content/stats/topfolderbyreports` URI.

GET http://<baseURL>/biprws/bionbi/content/stats/topfolderbyreports

If you use this API to make a `GET` request to `/biprws/bionbi/content/stats/topfolderbyreports` URI. You receive the following response:

Response:

- In response, you see the list of folders with maximum number of report documents, with highest count of report documents listed on top along with its details like: id, cuid, title, description, report_count, type, parent_id details.
-
```
<feed>
<id>tag:sap.com,2010:bip-rs/bionbi</id>
<updated>2015-08-19T06:19:30.465Z</updated>
<title type="text">Metrics for content Statistics - Folder with highest reports</
title>
<entry>
```

```
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">AYfaKZs_CdNEsztCbUVNGLU</attr>
<attr name="report_count" type="int32">11</attr>
<attr name="parent_id" type="int32">5668</attr>
<attr name="description" type="string" />
<attr name="id" type="int32">5758</attr>
<attr name="title" type="string">Feature Samples</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">AYRMR1n5A5hGnXSy3tPmXTc</attr>
<attr name="report_count" type="int32">8</attr>
<attr name="parent_id" type="int32">5668</attr>
<attr name="description" type="string" />
<attr name="id" type="int32">5918</attr>
<attr name="title" type="string">Financial</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">AeN4lEu0h_tAtnPEjFYxwi8</attr>
<attr name="report_count" type="int32">6</attr>
<attr name="parent_id" type="int32">0</attr>
<attr name="description" type="string" />
<attr name="id" type="int32">5632</attr>
<attr name="title" type="string">Web Intelligence Samples</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">AbShpyV1zOlJntqMZ31WhAc</attr>
<attr name="report_count" type="int32">6</attr>
<attr name="parent_id" type="int32">5632</attr>
<attr name="description" type="string" />
<attr name="id" type="int32">5633</attr>
<attr name="title" type="string">Mobile Samples</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">AfpS1e1GVBZAkeuBlI4KR0I</attr>
<attr name="report_count" type="int32">3</attr>
<attr name="parent_id" type="int32">5668</attr>
<attr name="description" type="string" />
<attr name="id" type="int32">5669</attr>
<attr name="title" type="string">Demonstration</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">AS1oZEJAynpNjZIaZK2rc7g</attr>
<attr name="report_count" type="int32">1</attr>
<attr name="parent_id" type="int32">0</attr>
<attr name="description" type="string" />
<attr name="id" type="int32">578</attr>
<attr name="title" type="string">Auditing</attr>
```

```
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">AX0OyM93QQZBmCjz8zfCWyc</attr>
<attr name="report_count" type="int32">1</attr>
<attr name="parent_id" type="int32">4760</attr>
<attr name="description" type="string" />
<attr name="id" type="int32">4924</attr>
<attr name="title" type="string">Report Conversion Tool Audit Documents</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">Acu9FvxWBZ9Htt0_08a25b4</attr>
<attr name="report_count" type="int32">1</attr>
<attr name="parent_id" type="int32">0</attr>
<attr name="description" type="string" />
<attr name="id" type="int32">5656</attr>
<attr name="title" type="string">Monitoring Report Sample</attr>
 </attrs>
</content>
</entry>
 </feed>
```


URI, you can perform following operation:

- Listing top `N` folders that have maximum number of report documents

Listing Top `N` Folders That Have Maximum Number of Report Documents:

If you want to list top N folders that have maximum number of report documents, use `http://<baseURL>/biprws/bionbi/content/stats/topfolderbyreports?topN=<Top_Value>`

> **Note**
>
> By default all folders are listed.

Parameter Description

Parameter Name	Value	Description
topN	`<Top_Value>`	Lists top `1 to N` number of folders that have maximum number of report documents.

9.3.4 Statistics Inbox by Unread

To list the inboxes having maximum number of unread report documents, with highest unread report documents listed on top, use `http://<baseURL>/biprws/bionbi/content/stats/topinboxbyunread` URI.

GET http://<baseURL>/biprws/bionbi/content/stats/topinboxbyunread

If you use this API to make a `GET` request to `/biprws/bionbi/content/stats/topinboxbyunread` URI. You receive the following response:

Response:

- In response, you see the list of inboxes with maximum number of unread report documents, with highest count of report documents listed on top along with its details like: id, cuid, title, description, report_count, type, parent_id details.
-
```
<feed>
<id>tag:sap.com,2010:bip-rs/bionbi</id>
<updated>2015-08-19T06:21:46.163Z</updated>
<title type="text">Metrics for Content Statistics - Inbox with Highest Unread
reports</title>
<entry>
<content type="application/xml">
 <attrs>
<attr name="unread_report_count" type="int32">2</attr>
<attr name="cuid" type="string">CQEwAAAA9FZpPn7XfU2Ust.ltEwspg</attr>
<attr name="parent_id" type="int32">48</antr>
<attr name="description" type="string" />
<attr name="read_report_count" type="int32">0</attr>
<attr name="id" type="int32">876</attr>
<attr name="title" type="string">Administrator</attr>
 </attrs>
</content>
</entry>
 </feed>
```

Using this URI, you can perform following operation:

- Listing top `N` inboxes having maximum number of unread report documents

Listing Top `N` Inboxes Having Maximum Number of Unread Report Documents:

If you want to list top N inboxes that have maximum number of unread report documents, use `http://<baseURL>/biprws/bionbi/content/stats/topinboxbyunread?topN=<Top_Value>`

> **Note**
>
> By default all inboxes are listed.

Parameter Description

Parameter Name	Value	Description
topN	`<Top_Value>`	Lists top `1 to N` number of inboxes that have maximum number of unread report documents.

9.3.5 Statistics Universe by Reports

To list the Universes having maximum number of report documents created on it, with highest report documents listed on top, use `http://<baseURL>biprws/bionbi/content/stats/topunvbyreports` URI.

GET http://<baseURL>biprws/bionbi/content/stats/topunvbyreports

If you use this API to make a `GET` request to `biprws/bionbi/content/stats/topunvbyreports` URI. You receive the following response:

Response:

- In response, you see the list of Universes with maximum number of report documents created on it, with highest count of report documents listed on top along with its details like: id, cuid, title, description, report_count, type, parent_id details.
-
```
<feed>
<id>tag:sap.com,2010:bip-rs/bionbi</id>
<updated>2015-08-19T06:22:41.097Z</updated>
<title type="text">Metrics for Content Statistics - Universe with Highest
Reports</title>
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">AX3cE9nWhMBLtXyWJ56OoDQ</attr>
<attr name="report_count" type="int32">4</attr>
<attr name="parent_id" type="int32">601</attr>
<attr name="description" type="string">eFashion retail Data Warehouse created 14
Oct 1998. Updated 3 April 2002. 89,000+ row fact table. Version 6.0. Updated 18
June 2014 with the efashion connection (MS Access 2007) by XAVIER</attr>
<attr name="id" type="int32">5643</attr>
<attr name="title" type="string">eFashion</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">AZtmGeZqCllIugfputlCuho</attr>
<attr name="report_count" type="int32">2</attr>
<attr name="parent_id" type="int32">5645</attr>
<attr name="description" type="string">eFashion retail Data Warehouse dated 14
Oct 2007. 89,000+ row fact table. Version 13</attr>
<attr name="id" type="int32">5646</attr>
<attr name="title" type="string">eFashion</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">AZZrE5CfJd9EjCBBvwuqgVA</attr>
<attr name="report_count" type="int32">1</attr>
<attr name="parent_id" type="int32">5655</attr>
<attr name="description" type="string" />
<attr name="id" type="int32">5658</attr>
<attr name="title" type="string">Monitoring TrendData Universe</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="cuid" type="string">AVHuOHKBm5lFkr3KcJkfk_0</attr>
<attr name="report_count" type="int32">1</attr>
<attr name="parent_id" type="int32">4724</attr>
<attr name="description" type="string" />
<attr name="id" type="int32">5664
<attr name="title" type="string">Report Conversion Tool Audit Universe</attr>
 </attrs>
</content>
```

```
</entry>
 </feed>
```

Using this URI, you can perform following operation:

- Listing top N Universes having maximum number of report documents created

Listing Top N Universes Having Maximum Number of Report Documents Created:

If you want to list top N Universes that have maximum number of report documents created on it, use `http://<baseURL>biprws/bionbi/content/stats/topunvbyreports?topN=<Top_Value>`

> **Note**
>
> By default all Universes are listed.

Parameter Description

Parameter Name	Value	Description
topN	`<Top_Value>`	Lists top `1 to N` number of Universes that have maximum number of report documents created.

9.4 Applications

This section provides API that helps you to view count and usage statistics of different reports on your system. The following are the list of APIs:

- Application Count [page 353]
- Application List [page 356]

9.4.1 Application Count

To obtain the number of distinct reports' count, use `http://<baseURL>/bionbi/application` URI.

GET http://<baseURL>/bionbi/application

If you use this API to make a `GET` request to `/bionbi/application` URI. You receive the following response:

This API returns limited list of SI_KINDs, see the report kind listed in the table:

Report Kind

Report Kind
Webi
CrystalReport
AAD.AnalysisApplication
XL.XcelsiusEnterprise
DataDiscovery
Xcelsius
DataDiscoveryAlbum
MDAnalysis
AO.Workbook
VISILums
LumsExtension
Hyperlink
Note
pQuery
AO.Presentation
FullClient

Response:

-
```
<feed>
<id>tag:sap.com,2010:bip-rs/bionbi</id>
<title type="text">Metrics for ReportCountByKindForPeriod</title>
<updated>2015-08-20T06:07:10.139Z</updated>
<author>
<name>@win-2k8r2-x64:6400</name>
</author>
<entry>
<content type="application/xml">
 <attrs>
<attr name="KIND" type="string">Web Intelligence</attr>
<attr name="REPORT_COUNT" type="int32">5823</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="KIND" type="string">Crystal Report</attr>
<attr name="REPORT_COUNT" type="int32">5823</attr>
 </attrs>
</content>
```

```
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="KIND" type="string">Publication</attr>
<attr name="REPORT_COUNT" type="int32">3</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="KIND" type="string">Hyperlink</attr>
<attr name="REPORT_COUNT" type="int32">1</attr>
 </attrs>
</content>
</entry>
 </feed>
```

Using this URI, you can perform the following operation:

- Listing top N reports' count
- Sorting reports on the basis of count
- Listing reports created during a particular duration

Listing `top N` Reports' Count

You can list top N reports by using the following URI:

`http://10.160.212.35:6405/biprws/bionbi/content/application?topN=<Top_Value>`

> **Note**
>
> By default count for all reports are listed.

Parameter Description

Parameter Name	Value	Description
topN	`<Top_Value>`	Lists top `1 to N` reports that have maximum count.

Sorting Reports on the Basis of Count

Sorting of reports is based on the number of distinct reports in your system.

Using `http://<baseURL>/bionbi/application?sortType=<Sort_Type_value>` URI, you can perform sorting in ascending or descending order.

Following table gives you the parameters description:

Parameter Description

Parameter Name	Value	Description
sortType	`<Sort_Type_value>`	This parameter either takes the value `ASC`(Ascending) or `DESC`(Descending) as its value. **Note** By default `<sortType>` is set to DESC.

For example:`http://<baseURL>/bionbi/application?sortType=DESC`, displays the reports' count for distinct reports in descending order, based on the count

Listing Reports Created During a Particular Duration

If you want to know the distinct reports' count, use the query parameters `<startDate><endDate>`in the URI as follows:

> **Note**
>
> The provided date and time refers to UTC time zone.

```
http://<baseURL>/bionbi/application?startDate=<MM/DD/YYYY HH:MM:SS>&endDate=<MM/DD/
YYYY HH:MM:SS>
```

Parameter Description

Parameter Name	Value	Description
startDate	`<MM/DD/YYYY HH:MM:SS>`	This parameter is used to specify the start date in MM/DD/YYYY HH:MM:SS format.
endDate	`<MM/DD/YYYY HH:MM:SS>`	This parameter is used to specify the end date in MM/DD/YYYY HH:MM:SS format.

For example: `http://<baseURL>/bionbi/application?startDate=06/12/2015 00:00:00&endDate=07/15/2015 00:00:00` , displays the reports' count created between June 12th, 2015 and July 15th, 2015 for UTC time zone.

> **Note**
>
> - By default, `<startDate>` is 1/1/1900 00:00:00
> - By default,`<endDate>`is 1/1/2900 00:00:00

9.4.2 Application List

To obtain the number of users who scheduled, ran or viewed the report at a particular duration, use `http://<baseURL>/bionbi/application/list` URI.

GET http://<baseURL>/bionbi/application/list

If you use this API to make a `GET` request to `/bionbi/application/list` URI. You receive the following response:

Response:

- Note

  In response, you also see the navigation link details, which you can use to navigate between pages.

```
<feed>
<id>tag:sap.com,2010:bip-rs/bionbi</id>
<updated>2015-08-20T06:08:22.315Z</updated>
<title type="text">Metrics for ApplicationListForReportKind</title>
<link href="http://10.160.209.144:6405/biprws/bionbi/application/list?
page=1&pageSize=50&startDate=01/01/1900&endDate=01/01/2900&sortType=DESC"
rel="self" />
<link href="http://10.160.209.144:6405/biprws/bionbi/application/list?
page=1&pageSize=50&startDate=01/01/1900&endDate=01/01/2900&sortType=DESC"
rel="first" />
<link href="http://10.160.209.144:6405/biprws/bionbi/application/list?
page=1&pageSize=50&startDate=01/01/1900&endDate=01/01/2900&sortType=DESC"
rel="last" />
<entry>
<content type="application/xml">
 <attrs>
<attr name="KIND" type="string">Web Intelligence</attr>
<attr name="REPORT_COUNT" type="int32">5823</attr>
<attr name="USER_COUNT" type="string">1</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="KIND" type="string">Crystal Report</attr>
<attr name="REPORT_COUNT" type="int32">5823</attr>
<attr name="USER_COUNT" type="string">1</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="KIND" type="string">Publication</attr>
<attr name="REPORT_COUNT" type="int32">3</attr>
<attr name="USER_COUNT" type="string">1</attr>
 </attrs>
</content>
</entry>
<entry>
<content type="application/xml">
 <attrs>
<attr name="KIND" type="string">Hyperlink</attr>
<attr name="REPORT_COUNT" type="int32">1</attr>
<attr name="USER_COUNT" type="int32">0</attr>
 </attrs>
</content>
</entry>
 </feed>
```

Using this URI, you can perform the following operation:

- Listing top N disctinct reports with their usage statistics
- Sorting reports on the basis of count
- Listing reports and their usage statistics created during a particular duration
- Listing usage statistics Page by Page

Listing `top N` with Their Usage Statistics

You can list top N reports by using the following URI:

```
http://10.160.212.35:6405/biprws/bionbi/content/application?topN=<Top_Value>
```

> **Note**
>
> By default count for all reports are listed.

Parameter Description

Parameter Name	Value	Description
topN	`<Top_Value>`	Lists top `1 to N` reports that have maximum count.

Sorting Reports on the Basis of Count

Sorting of reports is based on the number of distinct reports in your system.

Using `http://<baseURL>/bionbi/application?sortType=<Sort_Type_value>` URI, you can perform sorting in ascending or descending order.

Following table gives you the parameters description:

Parameter Description

Parameter Name	Value	Description
sortType	`<Sort_Type_value>`	This parameter either takes the value `ASC`(Ascending) or `DESC`(Descending) as its value. **Note** By default `<sortType>` is set to DESC.

For example:`http://<baseURL>/bionbi/application?sortType=DESC`, displays the reports' count for distinct reports in descending order, based on the count.

Listing Reports and Their Usage Statistics Created Within a Particular Duration

If you want to know the distinct reports' count, use the query parameters `<startDate><endDate>`in the URI as follows:

> **Note**
>
> The provided date and time refers to UTC time zone.

```
http://<baseURL>/bionbi/application?startDate=<MM/DD/YYYY HH:MM:SS>&endDate=<MM/DD/
YYYY HH:MM:SS>
```

Parameter Description

Parameter Name	Value	Description
startDate	`<MM/DD/YYYY HH:MM:SS>`	This parameter is used to specify the start date in MM/DD/YYYY HH:MM:SS format.
end Date	`<MM/DD/YYYY HH:MM:SS>`	This parameter is used to specify the end date in MM/DD/YYYY HH:MM:SS format.

For example: `http://<baseURL>/bionbi/application?startDate=06/12/2015 00:00:00&endDate=07/15/2015 00:00:00` , displays the reports' count created between June 12th, 2015 and July 15th, 2015 for UTC time zone.

> **Note**
>
> - By default, `<startDate>` is 1/1/1900 00:00:00
> - By default,`<endDate>`is 1/1/2900 00:00:00

Listing Usage Statistics Page by Page:

You can either use the navigation link obtained in the response or you can manually enter `page` and `pageSize` parameter details as shown below:

To list report usage statistics page by page use the following URI:

```
http://<baseURL>/bionbi/application/list?page=<Page_no>&pageSize=<Max_Page_Limit>
```

Following table gives you the description for the parameters `page` and `pageSize`:

Parameter Description

Parameter Name	Value	Description
`page`	`<Page_no>`	The parameter `page` specifies the page number.
`pageSize`	`<Max_Page_Limit>`	The parameter `pageSize` specifies the maximum limit of a page. **Note** By default, `pageSize=50`

For example 1: Use `http://<baseURL>/bionbi/application/list?page=1&pageSize=10`, lists first 10 reports usage statistics in one page.

For example 2: Use `http://<baseURL>/bionbi/application/list?page=2&pageSize=10`, lists 11 through 20 reports usage statistics in the second page.

10 Appendix

10.1 Appendix A - RWS error messages summary, categorized

The following table lists RESTful Web Services error codes. Bracketed items indicated as `{ * insert resource name here * }`, in this table are replaced by the relevant resource name or value in the error message.

For more information about RWS and any other BI platform error messages, see the BusinessObjects XI *Error Messages Explained* guide.

The following table summarizes the errors organized by category.

Category	<message>
WebApplicationMapper	
RWS 00002	General server error.
RWS 00003	Client input error.
NoAccessException	
RWS 00004	Forbidden.
RSPluginException	
RWS 00006	Unable to create service. See server logs for details.
RWS 00007	Unknown error occurred while invoking service. See server logs for details.
RWS 00010	Resource not supported for the requested object.
InvalidEntSessionException	
RWS 00008	The HTTP header does not contain the X-SAP-LogonToken attribute.
RWS 00011	Invalid session token timeout value: { * insert resource name here * }.
RWS 00016	The server session is not available from the PJS service bean.
RWS 00076	Logon may not proceed because a session is already associated with this request.
RWS 00079	Please validate your input.
NotFoundException	
RWS 00005	Not Found.

Category	<message>
RWS 00009	Resource not found: { * insert resource name here * }.
RWS 00012	Info object with ID { * insert resource name here * } not found.
RWS 00015	No relationship named { * insert resource name here * }.
DuplicateException	
RWS 00013	Duplicate Object.
RWS 00051	A duplicate { * insert resource name here * } instance was created.
CodecException	
RWS 00017	Encode failure.
RWS 00018	{ * insert resource name here * } is NULL.
RWS 00019	Illegal Argument: { * insert resource name here * }.
RWS 00020	Cannot serialize value of type { * insert resource name here * }.
RWS 00021	Unterminated string.
RWS 00022	Malformed date: { * insert resource name here * }.
RWS 00023	Malformed time: { * insert resource name here * }.
RWS 00024	Malformed datetime: { * insert resource name here * }.
RWS 00025	Cannot deserialize value of type { * insert resource name here * }.
RWS 00026	Cannot get the attribute name. The name is either null or empty.
<reserved>	
RWS 00001	<reserved>
RWS 00014	<reserved>
RWS 00027	<reserved>
RWS 00028	<reserved>
RWS 00029	<reserved>
RWS 00030	<reserved>
ModelException	
RWS 00031	Model error.

Category	<message>
RWS 00032	No setter.
RWS 00033	Parameters must not be used with this getter command: { * insert resource name here * }.
RWS 00034	Setter must have exactly one parameter: { * insert resource name here * }.
RWS 00035	Setter { * insert resource name 1 here * } is not of the same type as getter { * insert resource name 2 here * }.
RWS 00036	Source: { * insert resource name 1 here * } + destination: { * insert resource name 2 here * }.
RWS 00037	Reference equality is not implemented.
RWS 00038	This use in hash-based collections is not implemented.
RWS 00039	Class { * insert resource name here * } is not a model class.
RWS 00040	Class { * insert resource name here * } is not a model class.
RWS 00041	Attribute ''{ * insert resource name 1 here * }'' cannot bind to two get (or set) methods: { * insert resource name 2 here * }, and { * insert resource name 3 here * }.)
RWS 00042	Model contains at least 1 write-only attribute. name: { * insert resource name 1 here * }, method: { * insert resource name 2 here * }.)
RWS 00043	No accessible constructor without parameters for class { * insert resource name here * }.)
RWS 00044	{ * insert resource name 1 here * } object is null for composition property { * insert resource name 2 here * }.
RWS 00045	Couldn't inject property ''{ * insert resource name 1 here * }'' to field { * insert resource name2 here * } of type { * insert resource name 3 here * }.
RWS 00046	Property name already exists: { * insert resource name here * }.
RWS 00047	GUID must not contain the path separator '/'.

Category	<message>
RWS 00048	No type for class { * insert resource name here * }.
RWS 00049	Empty filter.
RWS 00050	Filter may not use ''{ * insert resource name here * }'' in conjunction with any other filter strings.
RWS 00080	Cannot bind unknown attribute "{ * insert resource name 1 here * }'' to method ''{ * insert resource name 2 here * }".
WebApplicationExceptionMapper	
RWS 00052	Bad request. (RWS00052) Corresponds with HTTP Response Code 400. This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616. Consult section 10.4 of RFC 2616 for more guidance on resolving this error. Applies to RWS 00052 to RWS 00075.
RWS 00053	Unauthorized. (RWS00053) Corresponds with HTTP Response Code 401.
RWS 00054	Payment required. (RWS00054) Corresponds with HTTP Response Code 402.
RWS 00055	Forbidden. (RWS00055) Corresponds with HTTP Response Code 403.
RWS 00056	Not found. (RWS00056) Corresponds with HTTP Response Code 404.
RWS 00057	Method not allowed. (RWS00057) Corresponds with HTTP Response Code 405.
RWS 00058	Not acceptable. (RWS00058) Corresponds with HTTP Response Code 406.
RWS 00059	Proxy authentication required. (RWS00059) Corresponds with HTTP Response Code 407.
RWS 00060	Request timeout. (RWS00060) Corresponds with HTTP Response Code 408.
RWS 00061	Conflict. (RWS00061) Corresponds with HTTP Response Code 409.
RWS 00062	Gone. (RWS00062) Corresponds with HTTP Response Code 410.
RWS 00063	Length required. (RWS00063) Corresponds with HTTP Response Code 411.
RWS 00064	Precondition failed. (RWS00064) Corresponds with HTTP Response Code 412.
RWS 00065	Request entity too large. (RWS00065) Corresponds with HTTP Response Code 413.

Category	<message>
RWS 00066	Request-URI too long. (RWS00066) Corresponds with HTTP Response Code 414.
RWS 00067	Unsupported media type. (RWS00067) Corresponds with HTTP Response Code 415.
RWS 00068	Requested range not satisfiable. (RWS00068) Corresponds with HTTP Response Code 416.
RWS 00069	Expectation failed. (RWS00069) Corresponds with HTTP Response Code 417.
RWS 00070	Internal server error. (RWS00070) Corresponds with HTTP Response Code 500.
RWS 00071	Not implemented. (RWS00071) Corresponds with HTTP Response Code 501.
RWS 00072	Bad gateway. (RWS00072) Corresponds with HTTP Response Code 502.
RWS 00073	Service unavailable. (RWS00073) Corresponds with HTTP Response Code 503.
RWS 00074	Gateway timeout. (RWS00074) Corresponds with HTTP Response Code 504.
RWS 00075	HTTP version not supported. (RWS00075) Corresponds with HTTP Response Code 505.
CredentialException	
RWS 00077	The authentication scheme you have chosen is currently not supported.
RWS 00078	The credentials could not be decoded.

10.2 RESTful Web Services (RWS) Error Messages

RESTful Web Services error messages include the following:

Range	Category
RWS 00002 - RWS 00010	RESTful Web Services
RWS 000011 - RWS 000026	RESTful Web Services
RWS 000031 - RWS 000051	RESTful Web Services
RWS 00052 - RWS 00075	RESTful Web Services
RWS 000076 - RWS 000079	RESTful Web Services

10.2.1 RWS 00002 - RWS 00010

10.2.1.1 `General server error. (RWS 00002)`

Cause

An unknown error occurred in the BIP RESTful Web Service.

Action

Please check the server logs for more details.

10.2.1.2 `Client input error. (RWS 00003)`

Cause

There is an unknown error in the input of the client provided to the BIP RESTful Web Service.

Action

Please consult the documentation for the resource you're trying to call to determine if your input was indeed valid.

10.2.1.3 `Forbidden (RWS 00004)`

Cause

This resource may not be accessed.

Action

Verify you have the right permissions to access the resource.

10.2.1.4 `Not Found (RWS 00005)`

Cause

The specific resource could not be found. Either the resource does not exist or you do not have the permissions to view it.

Action

Verify that the URL you used was correct. If you're trying to view an InfoObject, use the Central Management Console (CMC) to verify that you have the right to view that object.

10.2.1.5 `Unable to create service. See server logs for details. (RWS 00006)`

Cause

The BIP RESTful Web Service was unable to create the requested service.

Action

Examine the JavaDoc for Constructor.newInstance. Cross check the cause of this exception with the exceptions thrown by Constructor.newInstance.

10.2.1.6 `Unknown error occurred while invoking service. See server logs for details. (RWS 00007)`

Cause

The BIP RESTful Web Service encountered an unknown error while invoking the service.

Action

Check the log of the Web Application Server containing the BIP RESTful Web Service to see more details.

10.2.1.7 `The HTTP header does not contain the X-SAP-LogonToken attribute. (RWS 00008)`

Cause

Access to the requested resources requires you to have been authenticated.

Action

Please pass in the X-SAP-LogonToken in the request's header. You may generate one using the logon resource.

10.2.1.8 `Resource not found: {0} (RWS 00009)`

Cause

The specific resource could not be found. Either the resource does not exist or you do not have the permissions to view it.

Action

Verify that the URL you used was correct. If you're trying to view an InfoObject, use the Central Management Console (CMC) to verify that you have the right to view that object.

10.2.1.9 `Resource not supported for the requested object. (RWS 00010)`

Cause

You attempted to access a resource for an InfoObject which was not supported. For example, this exception would be thrown when you try to access the Crystal Reports service for a Folder.

Action

Don't call this method on unsupported objects. Only visit links that are valid.

10.2.2 RWS 00011 - RWS 00026

10.2.2.1 `Invalid session token timeout value: {0}. (RWS 000011)`

Cause

A logon token could not be created because of an invalid setting in the BIP RESTful Web Service.

Action

Please contact your administrator to set an appropriate session token timeout value for the BIP RESTful Web Service in the Central Management Console (CMC).

10.2.2.2 `Info object with ID {0} not found. (RWS 000012)`

Cause

The InfoObject could not be found. If it's suppose to exist, have you verified that you have the permissions to view it?

Action

Use the Central Management Console (CMC) to verify that the InfoObject exists and that you have the right to view it.

10.2.2.3 `Duplicate Object (RWS 000013)`

Cause

A duplicate object was detected.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.2.4 `No relationship named {0}. (RWS 000015)`

Cause

The relationship could not be found on the InfoObject.

Action

Verify that the URL used was one genereated by the BIP RESTful WebService by visiting the root object. If the URL is indeed valid, have you checked your permissions to verify that you have the appropriate rights to view the relationship?

10.2.2.5 `The server session is not available from the PJS service bean. (RWS 000016)`

Cause

The Adaptive Processing Server has not passed a server session to the BIP RESTful Web Service.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.2.6 `Encode failure. (RWS 000017)`

Cause

The BIP RESTful Web Service uses a codec to encode objects into a user-readable format (e.g., XML). Unfortunately, it looks an encoding error occurred.

Action

Please check the logs for more details about the parameter that caused this error. If the problem remains unclear, please contact SAP BusinessObjects support for assistance.

10.2.2.7 `{0} is NULL. (RWS 000018)`

Cause

The BIP RESTful Web Service uses a codec to encode objects into a user-readable format (e.g., XML). Unfortunately, during its execution, it couldn't reference to a value.

Action

Please check the logs for more details about the parameter that caused this error. If the problem remains unclear, please contact SAP BusinessObjects support for assistance.

10.2.2.8 `Illegal Argument: {0} (RWS 000019)`

Cause

The BIP RESTful Web Service uses a codec to encode objects into a user-readable format (e.g., XML). Unfortunately, during its execution, it detected an illegal argument.

Action

Please check the logs for more details about the parameter that caused this error. If the problem remains unclear, please contact SAP BusinessObjects support for assistance.

10.2.2.9 `Cannot serialize value of type {0}. (RWS 000020)`

Cause

The BIP RESTful Web Service uses a codec to encode objects into a user-readable format (e.g., XML). We were unable to serialize a value.

Action

Please check the logs for more details about the parameter that caused this error. If the problem remains unclear, please contact SAP BusinessObjects support for assistance.

10.2.2.10 `Unterminated string. (RWS 000021)`

Cause

The BIP RESTful Web Service uses a codec to encode objects into a user-readable format (e.g., XML). It encountered an unterminated string.

Action

Please check the logs for more details about the parameter that caused this error. If the problem remains unclear, please contact SAP BusinessObjects support for assistance.

10.2.2.11 `Malformed date: {0}. (RWS 000022)`

Cause

The BIP RESTful Web Service was unable to encode/decode the date passed into it.

Action

Please check the logs for more details about the parameter that caused this error. If the problem remains unclear, please contact SAP BusinessObjects support for assistance.

10.2.2.12 `Malformed time: {0}. (RWS 000023)`

Cause

The BIP RESTful Web Service was unable to encode/decode the time passed into it.

Action

Please check the logs for more details about the parameter that caused this error. If the problem remains unclear, please contact SAP BusinessObjects support for assistance.

10.2.2.13 `Malformed datetime: {0}. (RWS 000024)`

Cause

The BIP RESTful Web Service was unable to encode/decode the date time passed into it.

Action

Please make sure the date time is in a format recognized by the ATOM standard (RFC 4287). Check the logs for more details about the parameter that caused this error. If the problem remains unclear, please contact SAP BusinessObjects support for assistance.

10.2.2.14 `Cannot deserialize value of type {0}. (RWS 000025)`

Cause

The BIP RESTful Web Service uses a codec to encode objects into a user-readable format (e.g., XML). Unfortunately, it looks a decoding error occurred.

Action

Please check the logs for more details about the parameter that caused this error. If the problem remains unclear, please contact SAP BusinessObjects support for assistance.

10.2.2.15 `Cannot get the attribute name. The name is either null or empty. (RWS 000026)`

Cause

The BIP RESTful Web Service uses a codec to encode objects into a user-readable format (e.g., XML). While reading/writing the user-readable format, a parser error occurred.

Action

Please check the logs for more details about the parameter that caused this error. If the problem remains unclear, please contact SAP BusinessObjects support for assistance.

10.2.3 RWS 00031 - RWS 00051

10.2.3.1 `Model error. (RWS 000031)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.2 `No setter. (RWS 000032)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.3 `Getter must not have parameters: {0}. (RWS 000033)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.4 `Setter must have exactly one parameter: {0}. (RWS 000034)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.5 `Setter {0} is not of the same type as getter {1}. (RWS 000035)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.6 `source: {0} + destination: {1}. (RWS 000036)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.7 `Reference equality is not implemented. (RWS 000037)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.8 `This use in hash-based collections is not implemented. (RWS 000038)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.9 `Class {0} is not a model class. (RWS 000039)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.10 `property '{0}' cannot bind to two fields: {1}, and {2}. (RWS 000040)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.11 `Attribute '{0}' cannot bind to two get (or set) methods: {1}, and {2}. (RWS 000041)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.12 `Model contains at least 1 write-only attribute. name: {0}, method: {1}. (RWS 000042)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.13 `No accessible constructor without parameters for class {0}. (RWS 000043)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.14 `{0} object is null for composition property {1}. (RWS 000044)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.15 `Couldn't inject property '{0}' to field {1} of type {2}. (RWS 000045)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.16 `Property name already exists: {0} (RWS 000046)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.17 `GUID must not contain the path separator '/' (RWS 000047)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.18 `No type for class {0} (RWS 000048)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.19 `Empty filter. (RWS 000049)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.20 `Filter may not use '{0}' in conjunction with any other filter strings. (RWS 000050)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.3.21 `A duplicate {0} instance was created. (RWS 000051)`

Cause

The BIP RESTful Web Service code has singleton objects to manage its daily operations. Strangely, a duplicate of a singleton object was created.

Action

This error should not be thrown in a customer environment. If you have verified that your installation is correct and hasn't been corrupted, please contact SAP BusinessObjects support for help resolving this issue.

10.2.4 RWS 00052 - RWS 00075

10.2.4.1 `Cannot process the request; the request could not be processesd by the server due to malformed syntax (RWS 00052)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please see section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.2 `You are not authorized to perform this request (RWS 00053)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please see section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.3 `Payment required (RWS 00054)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please see section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.4 `Error while performing the request; the server is unable to process the request; the request should not be repeated (RWS 00055)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please see section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.5 `Error while performing the request; the server is unable to find the match for the Requested URI (RWS 00056)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please consult section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.6 `Method not allowed; method specified in the Request-Line is not allowed for the resource identified by the Request-URI (RWS 00057)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please consult section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.7 `Request cannot be processed; the resource identified by the request is only capable of generating response entities which have content characteristics which is not acceptable according to the accept headers sent in the request (RWS 00058)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please consult section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.8 `Proxy authentication required (RWS 00059)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please consult section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.9 `Request timeout; request was not sent within the time the server was prepared to wait; try making the request again (RWS 00060)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please see section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.10 `Request not processed; the request could not be completed due to a conflict with the current state of the resource (RWS 00061)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please consult section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.11 `Request not processed; the requested resource is no longer available at the server and no forwarding address is known (RWS 00062)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please see section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.12 `Request not processed; request does not contain header Content-Length; try making the request by adding valid data in Content-Length header field that contains length of the message body in the request message (RWS 00063)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please see section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.13 `Request not processed; request does not contain header Content-Length; try making the request by adding valid data in Content-Length header field that contains length of the message body in the request message (RWS 00063)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please see section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.14 `Precondition failed (RWS 00064)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please consult section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.15 `Request entity too large (RWS 00065)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please consult section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.16 `Request-URI too long (RWS 00066)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please consult section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.17 `Unsupported media type (RWS 00067)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please consult section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.18 `Requested range not satisfiable (RWS 00068)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please consult section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.19 `Request failed; the server could not process the request given in an Expect request-header field or if the server is a proxy server, it is not guaranteed that the request could be processed by the next-hop server (RWS 00069)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

See section 10.4 of RFC 2616 for more guidance on resolving this error.

10.2.4.20 `Internal server error (RWS 00070)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please consult section 10.5 of RFC 2616 for more guidance on resolving this error.

10.2.4.21 `Request not processed; the server does not support the functionality required to fulfill the request (RWS 00071)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

See section 10.5 of RFC 2616 for more guidance on resolving this error.

10.2.4.22 `Request not processed; the server while performing as gateway or proxy, received an invalid (RWS 00072)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

See section 10.5 of RFC 2616 for more guidance on resolving this error.

10.2.4.23 `Service unavailable (RWS 00073)`

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please consult section 10.5 of RFC 2616 for more guidance on resolving this error.

10.2.4.24 Gateway timeout (RWS 00074)

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please consult section 10.5 of RFC 2616 for more guidance on resolving this error.

10.2.4.25 HTTP version not supported (RWS 00075)

Cause

This is a generic error message thrown by the BIP RESTful Web Service under the circumstances dictated by RFC 2616.

Action

Please consult section 10.5 of RFC 2616 for more guidance on resolving this error.

10.2.5 RWS 00076 - RWS 00080

10.2.5.1 Logon may not proceed because a session is already associated with this request. (RWS 000076)

Cause

You attempted to log onto the BIP RESTful Web Service while a session has already been associated with the request.

Action

Don't pass in a session to the BIP RESTful Web Service when you use the Logon resource.

10.2.5.2 `The authentication scheme you have chosen is currently not supported. (RWS 000077)`

Cause

The selected authentication scheme you have chosen is not supported by the BI Platform RESTful Web Service.

Action

Either pass in the credentials using the X-SAP-LogonToken mechanism or use HTTP BASIC authentication (see RFC 2617).

10.2.5.3 `The credentials could not be decoded. (RWS 000078)`

Cause

The credentials passed into the BI Platform RESTful Web Service could not be decoded.

Action

Make sure credentials are encoded correctly before using them. If you're using HTTP BAISC authentication, make sure they're encoded in the format specified by RFC 2617.

10.2.5.4 `Enter a valid input (RWS 000079)`

Cause

Please make sure the content of your request is formatted correctly and contains all the necessary fields.

Action

Re-send the request after you've verified that that content of your request is formatted correctly. Typically, you may use GET to determine what format the request should be in. You may also check the documentation for this information as well.

10.2.5.5 `Cannot bind unknown attribute {0} to method {1}. (RWS00080)`

Cause

The BIP RESTful Web Service contains invalid data in its binaries.

Action

Because this error message is rare in a fully installed environment, it may indicate a faulty or corrupted installation. If you have verified that your installation is correct and hasn't been corrupted, contact SAP BusinessObjects support for help resolving this issue.

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.
About the icons:

- Links with the icon : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
  - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
  - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon : You are leaving the documentation for that particular SAP product or service and are entering a SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.
The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Gender-Related Language

We try not to use gender-specific word forms and formulations. As appropriate for context and readability, SAP may use masculine word forms to refer to all genders.

go.sap.com/registration/contact.html

© 2018 SAP SE or an SAP affiliate company. All rights reserved.
No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company. The information contained herein may be changed without prior notice.
Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.
These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.
SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.
Please see https://www.sap.com/about/legal/trademark.html for additional trademark information and notices.