PUBLIC

SAP BusinessObjects Business Intelligence platform

Document Version: 4.3 Support Package 4 – 2023-12-07

SAP BusinessObjects RESTful Web Service SDK User Guide for Web Intelligence and the BI Semantic Layer

© 2024 SAP SE or an SAP affiliate company. All rights reserved.

THE BEST RUN SAP

Content

1 Document Version History. 10

2 REST API Quick Reference. 13

2.1 Configuration APIs. 13

2.2 Universe APIs. 14

2.3 Connection APIs. 14

2.4 Spreadsheet APIs. 15

2.5 Query APIs. 16

Query Refresh APIs. 16

Query Result APIs. 17

2.6 Document APIs. 18

Alerter APIs. 19

Custom Palette APIs. 20

Input Control APIs. 20

Merged Dimension APIs. 21

Style APIs. 22

Tracker APIs. 23

Variable APIs. 23

ZIP File APIs. 24

2.7 Document Lifecycle APIs. 24

2.8 Document Refresh APIs. 25

2.9 Document Scheduling APIs. 26

2.10 Report APIs. 26

Report Data Filter APIs. 28

Report Input Control APIs. 28

Report Drill APIs. 29

Report Drill Filter APIs. 30

2.11 Report Element APIs. 31

Report Element Axis APIs. 32

Report Element Custom Property APIs. 33

Report Element Data Filter APIs. 34

Report Element Calculation APIs. 35

Report Element Ranking APIs. 36

Report Element Sorting APIs. 36

2.12 Data Provider APIs. 37

Data Provider Change Source APIs. 38

Data Provider Refresh APIs. 39

2.13 Publication API. 39
2.14 Variant APIs. 40
2.15 SAP BW Connection and BEx Query APIs. 40
2.16 Search Resource API. 41
2.17 Comments APIs. 41
2.18 Session Security API. 42
2.19 Web Intelligence OData APIs. 43

3 Introduction to the RESTful Web Service SDK User Guide. 45
3.1 Audience. 45
3.2 Conventions in This Guide. 45
3.3 Key Tasks. 46
3.4 Use Cases. 46

4 Using the RESTful Web Service SDKs. 48
4.1 Default Base URLs. 50
4.2 To Log on to the BI platform. 51
4.3 Document State. 51
4.4 Getting Web Intelligence RESTful Web Service SDK Information. 52
4.5 Getting the Logged-in User Details. 54
4.6 Supporting Multiple Languages. 55
4.7 Dispensable Information in a Request Body. 55
4.8 HTTP Request Status. 57
4.9 Success and Error Messages. 58

5 Using the RESTful Web Service Samples. 61
5.1 About the BI Semantic Layer REST API Samples. 61
To Deploy the Samples with Eclipse. 62
5.2 About the Web Intelligence REST API Samples. 63
To Deploy the LCMBIAR File. 65
To Deploy the Samples with Eclipse. 65

6 Concepts and Schemas of the RESTful Web Service SDKs. 67
6.1 Report Structure. 67
6.2 Charts. 69
Chart Types. 69
Chart Response Body Schema. 70
6.3 Date and Time Formats. 104
6.4 Queries. 105
Query Specification Body Schema. 106
Query Options. 107
Result Objects. 108
Sort Objects. 109

Query Filters. . . . 110
Combined Queries. . . . 127
The OData Query Service. . . . 129
6.5 Parameters. . . . 132
Parameter Response Body Schemas. . . . 133
Lists of Values. . . . 136
Default Values. . . . 140
Previous Values. . . . 141
Incompatible Contexts. . . . 141
Answer Request Body Schemas. . . . 142
6.6 Change Source. . . . 145
Mapping Request and Response Body Schemas. . . . 147
Strategies. . . . 149
Compatibility Rules. . . . 150
6.7 Schedules. . . . 152
6.8 Universes. . . . 157
6.9 Object Full Paths. . . . 161
6.10 User Rights. . . . 162
6.11 Getting the Web Intelligence Security Rights for the Current User. . . . 163

7 BI Semantic Layer REST API Reference. . . . 166
7.1 Managing Universes with the BI Semantic Layer RESTful Web Service SDK. . . . 166
Getting the List of the Universes (Semantic Layer). . . . 166
Getting the Details of a Universe (Semantic Layer). . . . 168
Getting the Business Views of a Universe. . . . 171
Getting the Query Capabilities of a Universe (Semantic Layer). . . . 174
Getting the Link Groups of a Universe. . . . 176
Getting the Object Parameters of a Universe. . . . 178
7.2 Managing Queries. . . . 179
Creating a Query. . . . 180
Getting the List of Queries. . . . 181
Getting the Details of a Query. . . . 182
Deleting a Query. . . . 183
7.3 Managing Parameters. . . . 184
Getting the List of Parameters. . . . 184
Responding to Parameters. . . . 187
Getting the Details of a Parameter. . . . 199
7.4 Managing Query Results. . . . 205
Accessing the OData Service. . . . 205
Getting the OData Flow Metadata. . . . 207
Getting the OData Flow Content. . . . 209
OData Content - Getting All Data. . . . 211

OData Content - Getting the Row Count. . . . 214
OData Content - Getting the First Row. . . . 215
OData Content - Getting Property Content. . . . 215
OData Content - Getting Property Raw Content. . . . 216
OData Content - Getting Content After Offset. . . . 216
OData Content - Getting the First N Rows. . . . 219
OData Content - Getting the First N Rows After Offset. . . . 220

8 Web Intelligence REST API Reference. . . . 223
8.1 Managing Documents. . . . 223
Creating a Document. . . . 224
Getting the List of Documents. . . . 226
Getting the Details of a Document. . . . 228
Getting a Document's Security Rights for the Current User. . . . 231
Saving a Document. . . . 233
Copying a Document. . . . 236
Purging a Document. . . . 237
Working with ZIP files. . . . 238
Exporting Documents. . . . 242
Updating the State of a Document. . . . 247
Deleting a Document. . . . 250
Working with Properties. . . . 251
Managing Styles, Formats, Fonts, Skins, and Charsets 254
Working with Color Palettes. . . . 276
Managing Functions, Operators, and Variables. . . . 284
Managing Links. . . . 295
Managing Alerters and Tracker Settings. . . . 301
Managing Attachments. . . . 318
Working with Snapshots. . . . 321
Working with Occurrences. . . . 325
Working with Referenced Cells. . . . 331
Adding a Cache Entry to a Document. . . . 337
8.2 Managing Reports. . . . 339
Creating a Report. . . . 339
Getting the List of Reports. . . . 341
Getting the Details of a Report. . . . 342
Moving a Report. . . . 344
Copying a Report. . . . 345
Updating the Properties of a Report. . . . 346
Deleting a Report. . . . 347
Getting the Map of a Report. . . . 348
Exporting Reports. . . . 349

Shared Element APIs. . . . 359
Updating the Structure of a Report. . . . 360
Managing Data Filters in Reports. . . . 365
Getting the Structure of a Report. . . . 371
Managing Report Comments. . . . 372
8.3 Drilling on Report Data. . . . 376
Getting the Drill Mode. . . . 377
Updating the Drill Mode. . . . 377
Enabling the Query Drill. . . . 379
Disabling the Query Drill. . . . 380
Getting Information on the Drill Hierarchies. . . . 380
Getting the Free Drill Elements. . . . 382
Creating a Drill Filter. . . . 383
Getting the Drill Filters of a Report. . . . 384
Getting the Details of a Drill Filter. . . . 385
Updating a Drill Filter. . . . 386
Removing a Drill Filter. . . . 387
Performing a Drill. . . . 388
Making a Snapshot of a Report in Drill Mode. . . . 390
8.4 Managing Report Elements. . . . 390
Creating a Report Element. . . . 391
Getting the List of Report Elements. . . . 396
Getting the Details of a Report Element. . . . 400
Getting the Background of a Report Element. . . . 411
Getting the Datapaths of a Report Element. . . . 412
Getting the Dataset of a Report Element. . . . 416
Getting the Details of a Hierarchy Member. . . . 422
Updating a Report Element. . . . 423
Inserting a Row or Column into a Table. . . . 429
Merging or Splitting Cells of a Table. . . . 430
Setting a Table Cell as Section. . . . 431
Exporting a Report Element. . . . 432
Deleting a Report Element. . . . 438
Ordering a Report Element. . . . 439
Working with Axes. . . . 440
Working with Rankings. . . . 464
Working with Calculations. . . . 468
Working with Custom Properties. . . . 473
Managing Data Filters in Report Elements. . . . 478
Working with Sorts. . . . 484
Managing Report Element Comments. . . . 493

8.5 Working With Input Controls. . . . 499
Working with Document Input Controls. . . . 500
Working with Report Input Controls. . . . 525
8.6 Managing Data Providers. . . . 554
Getting the List of Data Providers. . . . 555
Getting the Details of a Data Provider. . . . 557
Adding a Data Provider. . . . 560
Updating a Data Provider. . . . 563
Moving a Data Provider in a Document 571
Deleting a Data Provider. . . . 572
Changing the Data Providers. . . . 573
Getting the Flow Count of a Data Provider 583
Getting the Details of a Flow. . . . 584
Getting the Samples of a Flow. . . . 586
Getting the Query Plan. . . . 587
Updating the Query Plan. . . . 590
Getting the Query Specification. . . . 592
Updating the Query Specification. . . . 593
Getting the Definition of an Object. . . . 595
Getting the List of Values of a Data Object 596
Getting the List of Values Under a Specific Data Object Value. . . . 598
8.7 Managing Personal Data Providers. . . . 600
Uploading a Microsoft Excel File to the CMS Repository. . . . 600
Getting the List of Microsoft Excel Files. . . . 602
Getting the Details of a Microsoft Excel File. . . . 603
Updating a Microsoft Excel File to the CMS Repository. . . . 605
Deleting a Microsoft Excel File. . . . 606
8.8 Managing SAP BW Connections and BEx Queries. . . . 607
Getting the List of SAP BW Connections. . . . 607
Getting the Details of an SAP BW Connection. . . . 609
Browsing the Details of an SAP BW connection. . . . 610
Getting the Outline of a BEx Query. . . . 612
Getting the Capabilities of a BEx Query. . . . 614
8.9 Managing Connections for Free-Hand SQL Data Providers. . . . 616
Getting the List of Connections. . . . 616
Getting the Details of a Connection. . . . 618
8.10 Refreshing Documents. . . . 619
Getting the Refresh Parameters of a Document. . . . 619
Refreshing a Document. . . . 626
Cancelling the Refresh of a Document. . . . 647
Getting the Details of a Parameter. . . . 649

Working with Variants. 653
8.11 Refreshing Data Providers. 664
Getting the Refresh Parameters of a Data Provider. 665
Refreshing a Data Provider. 666
8.12 Scheduling Documents. 667
Getting the List of Schedules. 667
Getting the Details of a Schedule. 669
Adding a Schedule. 671
Deleting a Schedule. 676
8.13 Managing Publications. 677
Getting the Details of a Publication. 677
8.14 Managing Shared Elements. 679
Getting the List of Shared Elements. 679
Getting the Details of a Shared Element. 681
Editing a Shared Element. 683
Deleting a Shared Element. 684
Working with Shared Elements of a Document. 684
Working with Linked Report Elements. 688
8.15 Searching for Resources. 691
Getting Resources. 691
8.16 Managing Universes with the Web Intelligence RESTful Web Service SDK. 695
Getting the List of Universes (Web Intelligence). 695
Getting the Details of a Universe (Web Intelligence). 696
Getting the Query Capabilities of a Universe (Web Intelligence). 700
8.17 Getting Document Content Through OData. 702
Getting a Document's Cubes and Report Elements. 703
Getting a Document's Metadata. 703
Getting a Document's Cube Dataset. 704
Getting a Document's Report Element Dataset. 704
Getting a Schedule's Last Instance's Cubes and Report Elements. 705
Getting a Schedule's Last Instance Metadata. 705
Getting a Schedule's Last Instance's Cube Dataset. 706
Getting a Schedule's Last Instance's Report Element Dataset. 706
Getting a Report Element Data Flow Name (Deprecated). 707
Getting a Report Element Metadata (Deprecated). 708
Getting a Report Element Data (Deprecated). 708
Specifying the request. 709

9 REST API Workflows. 711
9.1 Running Queries. 711
9.2 Creating a Web Intelligence Document. 713
9.3 Listing Documents. 714

9.4 Setting Parameters and Exporting to PDF. 715
9.5 Setting Parameters and Scheduling Now. 716
9.6 Drilling into a Report. 717

1 Document Version History

The following tables provide an overview of the document changes.

SAP BusinessObjects RESTful Web Service SDK for Web Intelligence and the BI Semantic Layer 4.3 SP4 (December 2023)
Changes in the Web Intelligence OData API: • New parameter to refresh the document, see Specifying the request [page 709]. • New parameter to answer document's prompts, see Specifying the request [page 709].

SAP BusinessObjects RESTful Web Service SDK for Web Intelligence and the BI Semantic Layer 4.3 SP3 (December 2022)
Changes in the Web Intelligence OData API: • New base root URL for document • New base root URL schedule's last instance • New path to get cube's dataset • New path to get report element dataset • New supported operators

SAP BusinessObjects RESTful Web Service SDK for Web Intelligence and the BI Semantic Layer 4.3 SP2 (December 2021)
Changes in the BI Semantic Layer REST API output format: • URLs in XML headers and schemes reference doc-oasis declarations (http://docs.oasis-open.org instead of http://schemas.microsoft.com) • XML tags previously not prefixed are prefixed with "`a:`" New OData support in the Web Intelligence REST APIs

SAP BusinessObjects RESTful Web Service SDK for Web Intelligence and the BI Semantic Layer 4.3 (June 2020)

New features supported in SAP BusinessObjects Web Intelligence RESTful Web Service SDK:

- Report element calculations:
  - Get all calculations associated to a report element, see XML tags previously not prefixed are prefixed with "Getting the Calculations of a Report Element [page 468].
  - Get details of a calculation, see Getting the Details of a Calculation [page 470].
  - Create calculations, see Creating a Calculation [page 471].
  - Delete calculations, see Deleting a Calculation [page 472]
- Publications
  - Get the details of a publication, see Getting the Details of a Publication [page 677].
- Referenced cells
  - XML tags previously not prGet the list of referenced cells in a document, see Getting the List of a Document Referenced Cells [page 331].
  - Get the definition of a referencd cell, see Getting the Definition of a Referenced Cell [page 333].
  - Create a referenced cell, see Adding a New Referenced Cell [page 334].
  - Edit a referenced cell, see Editing the Definition of a Referenced Cell [page 336].
  - Delete a referenced cell, see Deleting a Referenced Cell [page 337].

Enhancements to existing APIs:

- Documents
  - Purge a document, see Purging a Document [page 237].
- Search
  - Use new `limit` and `offset` parameters when searching for resources on the server, see Getting Resources [page 691].
- Path
  - In any call that returns a path located in a user's folder in the BI Platform repository, the root of the path is **My Folders**, and no more **My Favorites**. For more information, see Getting the Details of a Document [page 228]
- Input controls
  - Reorder report input controls, see Reordering Input Controls in a Report [page 549].
  - Reorder document input controls, see Reordering Input Controls in a Document [page 523].
- Report elements
  - Create custom number or datetime formats, see Creating Custom Number or Datetime Formats [page 257].
  - Set a table cell as a section, see Setting a Table Cell as Section [page 431].
  - Order a report element, see Ordering a Report Element [page 439].
  - Delete all breaks in a report element, see Deleting all the Breaks in a Report Element [page 452].
  - Get the list of alerters of a report element, see Getting the List of Alerters for a Report Element [page 304]
  - Edit all alerters of a report element, see Editing all Alerters of a Report Element [page 310]
  - Delete all alerter of a report element, see Deleting all Alerters of a Report Element [page 312]
  - Get the details of a hierarchy member, see Getting the Details of a Hierarchy Member [page 422].
- Report element axes
  - Update all the breaks on a table axis, see Updating all the Breaks on a Table's Axis [page 453].
  - Delete all the breaks on a table axis, see Deleting all the Breaks on a Table's Axis [page 455].
- Configuration
  - Get all the marker palettes, see Getting all the Marker Palettes [page 278].

SAP BusinessObjects RESTful Web Service SDK for Web Intelligence and the BI Semantic Layer 4.3 (June 2020)

- Get all the gradient orientations, see Getting the Gradient Orientations [page 273].
- Get all the locales, see Getting the Locales [page 259].
- Aggregate configuration calls, see Aggregating Configuration Calls [page 274].

2 REST API Quick Reference

An overview of the REST APIs for Web Intelligence and the BI Semantic Layer.

REST APIs are grouped by functional area with descriptions and links to more information.

2.1 Configuration APIs

Use these REST APIs to get information on default configuration details declared on the WACS server, such as the default formats or color palettes (`/configuration`).

The table below indicates the release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters: None

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Getting the Charsets [page 270]	GET	/configuration/charsets	4.0 SP6 | 4.1
Getting the Chart Types [page 271]	GET	/configuration/visualizations	4.1 SP1
Getting the Default Color Palettes [page 276]	GET	/configuration/palettes	4.1 SP1
Getting the Configuration Formats [page 254]	GET	/configuration/formats	4.0 SP6 | 4.1
Getting the Font Mappings [page 258]	GET	/configuration/fontmappings	4.0 SP6 | 4.1
Getting the Formula Engine Functions [page 284]	GET	/configuration/functions	4.0 SP6 | 4.1
Getting the Formula Engine Operators [page 285]	GET	/configuration/operators	4.0 SP6 | 4.1
Getting the Report Skins [page 260]	GET	/configuration/skins	4.0 SP6 | 4.1
Getting the Locales [page 259]	GET	/configuration/locales	4.3
Getting the Gradient Orientations [page 273]	GET	/configuration/gradientorientations	4.3
Aggregating Configuration Calls [page 274]	POST	/configuration	4.3

2.2 Universe APIs

Use these APIs to work with universes (`/universes`) either on the Web Intelligence side or on the BI Semantic Layer side.

The tables below indicate the 4.0, 4.1, and/or 4.2 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/sl/v1`

Path parameters: `<universeID>` is the universe identifier retrieved from the list of universes.

Action	Method	URL	Since
Getting the List of the Universes (Semantic Layer) [page 166]	GET	`/universes`	4.1 SP2
Getting the Details of a Universe (Semantic Layer) [page 168]	GET	`/universes/<universeID>`	4.1 SP2
Getting the Business Views of a Universe [page 171]	GET	`/universes/<universeID>/businessviews`	4.1 SP2
Getting the Query Capabilities of a Universe (Semantic Layer) [page 174]	GET	`/universes/<universeID>/capabilities`	4.1 SP2
Getting the Link Groups of a Universe [page 176]	GET	`/universes/<universeID>/linkgroups`	4.1 SP2
Getting the Object Parameters of a Universe [page 178]	GET	`/universes/<universeID>/prompts`	4.1 SP6

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameter: `<universeID>` is the universe identifier retrieved from the list of universes.

Action	Method	URL	Since
Getting the List of Universes (Web Intelligence) [page 695]	GET	`/universes`	4.0 SP5 | 4.1
Getting the Details of a Universe (Web Intelligence) [page 696]	GET	`/universes/<universeID>`	4.0 SP5 | 4.1
Getting the Query Capabilities of a Universe (Web Intelligence) [page 700]	GET	`/universes/<universeID>/capabilities`	4.0 SP6 | 4.1

2.3 Connection APIs

The REST APIs to work with connections stored in the CMS repository (`/connections`).

The table below indicates the 4.0 and/or 4.1 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<connectionID>`: connection identifier retrieved from the list of connections

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Getting the List of Connections [page 616]	GET	/connections	4.1 SP5
Getting the Details of a Connection [page 618]	GET	/connections/<connectionID>	4.1 SP5

2.4 Spreadsheet APIs

The REST APIs to work with Microsoft Excel spreadsheets (`/spreadsheets`).

The table below indicates the 4.0, 4.1, and/or 4.2 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<spreadsheetID>`: Microsoft Excel file identifier retrieved from the list of files

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Uploading a Microsoft Excel File to the CMS Repository [page 600]	POST	/spreadsheets	4.1 SP5
Getting the List of Microsoft Excel Files [page 602]	GET	/spreadsheets	4.1 SP5
Getting the Details of a Microsoft Excel File [page 603]	GET	/spreadsheets/<spreadsheetID>	4.1 SP5
Updating a Microsoft Excel File to the CMS Repository [page 605]	PUT	/spreadsheets/<spreadsheetID>	4.2
Deleting a Microsoft Excel File [page 606]	DELETE	/spreadsheets/<spreadsheetID>	4.1 SP6

2.5 Query APIs

You can use these APIs to query BI Semantic Layer universes (`/queries`).

The table below indicates the 4.0, 4.1, and/or 4.2 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/sl/v1`

Path parameters:

- `<queryID>`: query statement identifier retrieved from the list of queries

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Creating a Query [page 180]	POST	/queries	4.1 SP2
Deleting a Query [page 183]	DELETE	/queries/<queryID>	4.1 SP2
Getting the Details of a Query [page 182]	GET	/queries/<queryID>	4.1 SP2
Getting the List of Queries [page 181]	GET	/queries	4.1 SP2

2.5.1 Query Refresh APIs

These REST APIs allow you to get and refresh parameters defined in universes like contexts, @Prompts, and object parameters (`/queries/.../parameters/`).

The table below indicates the 4.0, 4.1 and/or 4.2 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/sl/v1`

Path parameters:

- `<queryID>`: query statement identifier retrieved from the list of queries
- `<parameterID>`: parameter identifier

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Getting the List of Parameters [page 184]	GET	/queries/<queryID>/parameters	4.1 SP3
Responding to Parameters [page 187]	PUT	/queries/<queryID>/parameters	4.1 SP3

Action	Method	URL	Since
Getting the Details of a Parameter [page 199]	GET| PUT	/queries/<queryID>/parameters/<parameterID>	4.2

2.5.2 Query Result APIs

You can use these APIs to manage the results of a query on a universe (/queries/.../data.svc/).

The table below indicates the 4.0, 4.1, and/or 4.2 release in which support for the API was introduced.

Base URL: http://<server_name>:6405/biprws/sl/v1

Path parameters:

- <queryID>: query statement identifier retrieved from the list of queries
- <flowName>: query flow name
- <rowIndex>: row index of the query flow
- <fieldName>: name of an object of the flow

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Accessing the OData Service [page 205]	GET	/queries/<queryID>/data.svc/	4.1 SP2
Getting the OData Flow Metadata [page 207]	GET	/queries/<queryID>/data.svc/$metadata	4.1 SP2
Getting the OData Flow Content [page 209]	GET	/queries/<queryID>/data.svc/<flowName>	4.1 SP2
OData Content - Getting the Row Count [page 214]	GET	/queries/<queryID>/data.svc/<flowName>/$count	4.1 SP2
OData Content - Getting the First Row [page 215]	GET	/queries/<queryID>/data.svc/<flowName>(<rowIndex>)	4.1 SP2
OData Content - Getting Property Content [page 215]	GET	/queries/<queryID>/data.svc/<flowName>(<rowIndex>)/<fieldName>	4.1 SP2
OData Content - Getting Property Raw Content [page 216]	GET	/queries/<queryID>/data.svc/<flowName>(<rowIndex>)/<fieldName>/$value	4.1 SP2
OData Content - Getting Content After Offset [page 216]	GET	/queries/<queryID>/data.svc/<flowName>?$skip=<offset>	4.1 SP2
OData Content - Getting the First N Rows [page 219]	GET	/queries/<queryID>/data.svc/<flowName>?$top=<limit>	4.1 SP2
OData Content - Getting the First N Rows After Offset [page 220]	GET	/queries/<queryID>/data.svc/<flowName>?$skip=<offset>&$top=<limit>	4.1 SP2

2.6 Document APIs

The REST APIs to manage Web Intelligence documents (`/documents`).

The table below indicates the release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<dataObjectID>`: data object identifier retrieved from the definition of the input control to which the data object is assigned

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Creating a Document [page 224]	POST	/documents	4.0 SP6 | 4.1
Copying a Document [page 236]			
Getting the List of Documents [page 226]	GET	/documents	4.0 SP5 | 4.1
Getting a Document's Security Rights for the Current User [page 231]	GET	/documents/<documentID>/rights	4.2 SP4
Getting the Details of a Document [page 228]	GET	/documents/<documentID>	4.0 SP5 | 4.1
Exporting a Document in Listing Mode [page 243]			
Exporting a Document as a Series of Pages [page 245]	GET	/documents/<documentID>/pages	4.0 SP5 | 4.1
Saving a Document [page 233]	POST	/documents/<documentID>	4.1 SP4
Deleting a Document [page 250]	DELETE	/documents/<documentID>	4.0 SP6 | 4.1
Purging a Document [page 237]	PUT	/documents/<documentID>	4.3
Adding an Attachment [page 318]	POST	/documents/<documentID>/attachments	4.0 SP6 | 4.1
Getting the List of Attachments [page 320]	GET	/documents/<documentID>/attachments	4.0 SP6 | 4.1
Adding a Cache Entry to a Document [page 337]	POST	/documents/<documentID>/cache	4.0 SP6 | 4.1
Getting the CSS of a Document [page 268]	GET	/documents/<documentID>/css	4.0 SP6 | 4.1
Updating the CSS of a Document [page 269]	PUT	/documents/<documentID>/css	4.0 SP6 | 4.1

Action	Method	URL	Since
Getting the List of Values of a Data Object [page 596]	GET	/documents/<documentID>/dataobjects/<dataObjectID>/lov	4.1 SP3
Getting the List of Values Under a Specific Data Object Value [page 598]	PUT	/documents/<documentID>/dataobjects/<dataObjectID>/lov	4.1 SP3
Getting the Definition of an Object [page 595]	GET	/documents/<documentID>/dataobjects/<dataobjectID>	4.3
Getting Custom Formats for Numbers [page 256]	GET	/documents/<documentID>/formats	4.0 SP6 | 4.1
Getting the Properties of a Document [page 251]	GET	/documents/<documentID>/properties	4.0 SP6 | 4.1
Updating the Properties of a Document [page 253]	PUT	/documents/<documentID>/properties	4.0 SP6 | 4.1

2.6.1 Alerter APIs

Use these REST APIs to create, edit, or delete an alerter attached to a document (`/documents/.../alerters`).

The table below indicates the release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<alerterID>`: alerter identifier retrieved from the list of alerters

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Adding an Alerter [page 306]	POST	/documents/<documentID>/alerters	4.0 SP6 | 4.1
Getting the List of Alerters [page 302]	GET	/documents/<documentID>/alerters	4.0 SP6 | 4.1
Getting the Details of an Alerter [page 303]	GET	/documents/<documentID>/alerters/<alerterID>	4.0 SP6 | 4.1
Getting the List of Alerters for a Report Element [page 304]	GET	/documents/<documentId>/report/<reportId>/elements/<elementId>/alerters/	4.3
Editing an Alerter [page 308]	PUT	/documents/<documentID>/alerters/<alerterID>	4.0 SP6 | 4.1

Action	Method	URL	Since
Editing all Alerters of a Report Element [page 310]	PUT	/documents/<documentId>/report/<reportId>/elements/<elementId>/alerters	4.3
Deleting an Alerter [page 311]	DELETE	/documents/<documentID>/alerters/<alerterID>	4.0 SP6 | 4.1
Deleting all Alerters of a Report Element [page 312]	DELETE	/documents/<documentId>/report/<reportId>/elements/<elementId>/alerters	4.3

2.6.2 Custom Palette APIs

Use these REST APIs to create, edit or delete a custom color palette attached to a document (`/documents/.../palettes`).

The table below indicates the 4.0, 4.1, and/or 4.2 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<paletteID>`: custom palette identifier retrieved from the list of palettes

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Creating a Custom Palette [page 281]	POST	/documents/<documentID>/palettes	4.1 SP5
Getting the List of Custom Palettes [page 279]	GET	/documents/<documentID>/palettes	4.1 SP5
Getting the Details of a Custom Palette [page 280]	GET	/documents/<documentID>/palettes/<paletteID>	4.1 SP5
Updating the Definition of a Custom Palette [page 283]	PUT	/documents/<documentID>/palettes/<paletteID>	4.1 SP5
Deleting a Custom Palette [page 283]	DELETE	/documents/<documentID>/palettes/<paletteID>	4.1 SP5

2.6.3 Input Control APIs

Use these REST APIs to work with document input controls (`/documents/.../inputcontrols`).

The table below indicates the 4.0, 4.1, and/or 4.2 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<inputControlID>`: input control identifier retrieved from the list of input controls

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Getting the Input Controls of a Document [page 500]	GET	/documents/<documentID>/inputcontrols	4.1 SP6
Moving an Input Control from a Report to a Document [page 524]	PUT	/documents/<documentID>/inputcontrols?fromId=<fromID>	4.1 SP6
Getting the Definition of a Document Input Control [page 507]	GET	/documents/<documentID>/inputcontrols/<inputControlID>	4.1 SP6
Updating the Definition of a Document Input Control [page 510]	PUT	/documents/<documentID>/inputcontrols/<inputControlID>	4.1 SP6
Adding an Input Control to a Document [page 514]	POST	/documents/<documentID>/inputcontrols	4.1 SP6
Deleting a Document Input Control [page 522]	DELETE	/documents/<documentID>/inputcontrols/<inputControlID>	4.1 SP6
Getting the Selection of a Document Input Control [page 517]	GET	/documents/<documentID>/inputcontrols/<inputControlID>/selection	4.1 SP6
Setting the Selection of a Document Input Control [page 519]	PUT	/documents/<documentID>/inputcontrols/<inputControlID>/selection	4.1 SP6
Deleting the Selection of a Document Input Control [page 520]	DELETE	/documents/<documentID>/inputcontrols/<inputControlID>/selection	4.1 SP6
Getting the List of Values of a Document Input Control [page 516]	GET	/documents/<documentID>/inputcontrols/<inputControlID>/lov	4.2 SP04
Getting the List of Values of a Document Input Control [page 516]	PUT	/documents/<documentID>/inputcontrols/<inputControlID>/lov	4.2 SP04

2.6.4 Merged Dimension APIs

Use these REST APIs to work with merged dimensions in Web Intelligence (`/documents/.../links`).

The table below indicates the 4.0, 4.1, and/or 4.2 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents

- `<linkID>`: link identifier retrieved from the list of links

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Adding a Link [page 298]	POST	/documents/<documentID>/links	4.0 SP6 | 4.1
Getting the Links of a Document [page 296]	GET	/documents/<documentID>/links	4.0 SP6 | 4.1
Getting the Details of a Link [page 297]	GET	/documents/<documentID>/links/ <linkID>	4.0 SP6 | 4.1
Editing a Link [page 299]	PUT	/documents/<documentID>/links/<linkID>	4.0 SP6 | 4.1
Deleting a Link [page 300]	DELETE	/documents/<documentID>/links/<linkID>	4.0 SP6 | 4.1

2.6.5 Style APIs

Use these REST APIs to define, edit, and delete styles in documents (`/documents/.../styles`).

The table below indicates the 4.0, 4.1, and/or 4.2 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<styleID>`: style identifier from the list of styles of the document

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Getting the Styles of a Document [page 262]	GET	/documents/<documentID>/styles	4.0 SP6 | 4.1
Adding a Style to a Document [page 264]	POST	/documents/<documentID>/styles	4.0 SP6 | 4.1
Getting the Details of a Style [page 263]	GET	/documents/<documentID>/styles/<styleID>	4.0 SP6 | 4.1
Updating a Style [page 266]	PUT	/documents/<documentID>/styles/<styleID>	4.0 SP6 | 4.1
Deleting a Style [page 267]	DELETE	/documents/<documentID>/styles/<styleID>	4.0 SP6 | 4.1

2.6.6 Tracker APIs

Use these REST APIs to work with tracker settings in documents (`/documents/.../tracker`).

The table below indicates the 4.0, 4.1, and/or 4.2 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Getting the Tracker Settings [page 313]	GET	`/documents/<documentID>/tracker`	4.0 SP6 | 4.1
Creating the Tracker Settings [page 314]	POST	`/documents/<documentID>/tracker`	4.0 SP6 | 4.1
Editing the Tracker Settings [page 316]	PUT	`/documents/<documentID>/tracker`	4.0 SP6 | 4.1
Deleting the Tracker Settings [page 317]	DELETE	`/documents/<documentID>/tracker`	4.0 SP6 | 4.1

2.6.7 Variable APIs

Use these REST APIs to define, edit, and delete variables in documents (`/documents/.../variables`).

The table below indicates the 4.0, 4.1, and/or 4.2 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<variableID>`: variable identifier retrieved from the list of variables of the document

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Getting the Variables of a Document [page 286]	GET	`/documents/<documentID>/variables`	4.0 SP6 | 4.1
Adding a Variable [page 289]	POST	`/documents/<documentID>/variables`	4.0 SP6 | 4.1
Getting the Definition of a Variable [page 288]	GET	`/documents/<documentID>/variables/<variableID>`	4.0 SP6 | 4.1

Action	Method	URL	Since
Editing a Variable [page 292]	PUT	/documents/<documentID>/variables/<variableID>	4.0 SP6 | 4.1
Deleting a Variable [page 294]	DELETE	/documents/<documentID>/variables/<variableID>	4.0 SP6 | 4.1

2.6.8 ZIP File APIs

Use these REST APIs to create and get ZIP files from documents (/documents/.../zips).

The table below indicates the 4.0, 4.1, and/or 4.2 release in which support for the API was introduced.

Base URL: http://<server_name>:6405/biprws/raylight/v1

Path parameters:

- <documentID>: document identifier retrieved from the list of documents
- <zipID>: ZIP file identifier obtained by creating the zip file

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Creating a ZIP File from a Document [page 238]	POST	/documents/<documentID>/zips?reportIds=<reportID>[,...]	4.1 SP3
Getting a Document as a ZIP file [page 240]	GET	/documents/<documentID>/zips/<zipID>	4.1 SP3
Deleting a ZIP File [page 241]	DELETE	/documents/<documentID>/zips/<zipID>	4.1 SP3

2.7 Document Lifecycle APIs

The REST APIs to manage occurrences and snapshots of Web Intelligence documents (/documents/.../occurrences/, /documents/.../snapshots/).

The table below indicates the 4.0, 4.1, and/or 4.2 release in which support for the API was introduced.

Base URL: http://<server_name>:6405/biprws/raylight/v1

Path parameters:

- <documentID> is the document identifier retrieved from the list of documents.
- <occurrenceID> is the occurrence identifier retrieved from the list of occurrences.
- <snapshotID> is the snapshot identifier retrieved from the list of snapshots.

Query parameter: `<snapshotID>` is the snapshot identifier retrieved from the list of snapshots.

Action	Method	URL	Since
Updating the State of a Document [page 247]	PUT	`/documents/<documentID>`	4.0 SP5 | 4.1
Getting the List of Occurrences [page 327]	GET	`/documents/<documentID> /occurrences`	4.2 SP3
Getting the Details of an Occurrence [page 328]	GET	`/documents/<documentID> /occurrences/ <occurrenceID>`	4.2 SP3
Creating an Occurrence [page 327]	POST	`/documents/<documentID> /occurrences`	4.2 SP3
Updating an Occurrence [page 329]	PUT	`/documents/<documentID> /occurrences/ <occurrenceID>`	4.2 SP3
Creating a Document Snapshot [page 321]	POST	`/documents/<documentID>/snapshots`	4.0 SP6 | 4.1
Getting the List of Snapshots [page 322]	GET	`/documents/<documentID>/snapshots`	4.0 SP6 | 4.1
Restoring a Document to a Snapshot [page 323]	PUT	`/documents/<documentID>/? snapshotId=<snapshotID>`	4.0 SP6 | 4.1
Deleting a Document Snapshot [page 324]	DELETE	`/documents/<documentID>/snapshots/ <snapshotID>`	4.2 SP3
Deleting All Snapshots [page 325]	DELETE	`/documents/<documentID>/snapshots`	4.2 SP3

2.8 Document Refresh APIs

Use these REST APIs to refresh parameter values in Web Intelligence documents (`/documents/.../parameters/`).

The table below indicates the 4.0, 4.1, and/or 4.2 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<parameterID>`: parameter identifier retrieved from the list of refresh parameters

Query parameters: see the detailed description of the related API.

Action	Method	Request	Since
Getting the Refresh Parameters of a Document [page 619]	GET	/documents/<documentID>/parameters	4.0 SP5 | 4.1
Refreshing a Document [page 626]	PUT	/documents/<documentID>/parameters	4.0 SP5 | 4.1
Getting the Details of a Parameter [page 649]	GET| PUT	/documents/<documentID>/parameters/<parameterID>	4.2
Cancelling the Refresh of a Document [page 647]	PUT	/documents/<documentID>/parameters/execution?cancel=<mode>	4.0 SP6 | 4.1

2.9 Document Scheduling APIs

The REST APIs to schedule Web Intelligence documents for specific destinations and formats (`/documents/.../schedules/`).

The table below indicates the 4.0, 4.1, and/or 4.2 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<scheduleID>`: schedule identifier retrieved from the list of schedules

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Adding a Schedule [page 671]	POST	/documents/<documentID>/schedules	4.0 SP5 | 4.1
Getting the List of Schedules [page 667]	GET	/documents/<documentID>/schedules	4.0 SP5 | 4.1
Getting the Details of a Schedule [page 669]	GET	/documents/<documentID>/schedules/<scheduleID>	4.0 SP5 | 4.1
Deleting a Schedule [page 676]	DELETE	/documents/<documentID>/schedules/<scheduleID>	4.0 SP5 | 4.1

2.10 Report APIs

The REST APIs to work with reports in Web Intelligence documents (`/documents/.../reports/`). Use them to create, export a report, and to update its structure.

The table below indicates the 4.0 and/or 4.1 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<reportID>`: report identifier retrieved from the list of reports
- `<pageIndex>`: number of the page to export

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Creating a Report [page 339]	POST	/documents/<documentID>/reports	4.0 SP6 | 4.1
Copying a Report [page 345]			
Getting the List of Reports [page 341]	GET	/documents/<documentID>/reports	4.0 SP5 | 4.1
Getting the Visible Comments of a Report [page 373]	GET	/documents/<documentID>/reports/<reportID>/pages/comments	4.2 SP4
Getting the Visible Comments of a Page Range of a Report [page 374]	GET	/documents/<documentID>/reports/<reportID>/pages/<pageRange>/comments	4.2 SP4
Getting all the Comments or Only the Visible Comments of a Report Element [page 493]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>/comments	4.2 SP4
Getting a Specific Comment of a Report Element [page 495]	GET	GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/comments/<commentID>	4.2 SP4
Getting the Details of a Report [page 342]	GET	/documents/<documentID>/reports/<reportID>	4.0 SP5 | 4.1
Exporting a Report in Listing Mode [page 350]			
Exporting a Report as a Series of Pages [page 353]	GET	/documents/<documentID>/reports/<reportID>/pages	4.0 SP6 | 4.1
Exporting a Page or Range of Pages of a Report [page 355]	GET	/documents/<documentID>/reports/<reportID>/pages/<pageIndex>	4.0 SP6 | 4.1
Moving a Report [page 344]	POST	/documents/<documentID>/reports?fromId=<fromID>&toId=<toID>	4.0 SP6 | 4.1
Updating the Properties of a Report [page 346]	PUT	/documents/<documentID>/reports/<reportID>	4.0 SP6 | 4.1
Deleting a Report [page 347]	DELETE	/documents/<documentID>/reports/<reportID>	4.0 SP6 | 4.1
Getting the Map of a Report [page 348]	GET	/documents/<documentID>/reports/<reportID>/map	4.0 SP6 | 4.1
Getting the Structure of a Report [page 371]	GET	/documents/<documentID>/reports/<reportID>/specification	4.0 SP6 | 4.1

Action	Method	URL	Since
Updating the Structure of a Report [page 360]	PUT	/documents/<documentID>/reports/<reportID>/specification	4.0 SP6 | 4.1

2.10.1 Report Data Filter APIs

These APIs allow you to manage data filters in reports (/documents/.../reports/.../datafilter/).

The table below indicates the 4.0 and/or 4.1 release in which support for the API was introduced.

Base URL: http://<server_name>:6405/biprws/raylight/v1

Path parameters:

- <documentID>: document identifier retrieved from the list of documents
- <reportID>: report identifier retrieved from the list of reports

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Creating a Data Filter [page 366]	POST	/documents/<documentID>/reports/<reportID>/datafilter	4.1 SP2
Getting the Details of a Data Filter [page 367]	GET	/documents/<documentID>/reports/<reportID>/datafilter	4.1 SP2
Updating a Data Filter [page 368]	PUT	/documents/<documentID>/reports/<reportID>/datafilter	4.1 SP2
Deleting a Data Filter [page 370]	DELETE	/documents/<documentID>/reports/<reportID>/datafilter	4.1 SP2

2.10.2 Report Input Control APIs

Use these APIs to create and edit input controls and their selected values in a report (/documents/.../reports/.../inputcontrols/).

The table below indicates the 4.0 and/or 4.1 release in which support for the API was introduced.

Base URL: http://<server_name>:6405/biprws/raylight/v1

Path parameters:

- <documentID>: document identifier retrieved from the list of documents
- <reportID>: report identifier retrieved from the list of reports
- <inputControlID>: input control identifier retrieved from the list of input controls

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Getting the Input Controls of a Report [page 529]	GET	/documents/<documentID>/reports/<reportID>/inputcontrols	4.1 SP3
Moving an Input Control from a Document to a Report [page 550]	PUT	/documents/<documentID>/reports/<reportID>/inputcontrols?fromId=<fromID>	4.1 SP6
Getting the Definition of an Input Control [page 533]	GET	/documents/<documentID>/reports/<reportID>/inputcontrols/<inputControlID>	4.1 SP3
Updating the Definition of an Input Control [page 539]	PUT	/documents/<documentID>/reports/<reportID>/inputcontrols/<inputControlID>	4.1 SP3
Adding an Input Control [page 543]	POST	/documents/<documentID>/reports/<reportID>/inputcontrols	4.1 SP3
Deleting an Input Control [page 545]	DELETE	/documents/<documentID>/reports/<reportID>/inputcontrols/<inputControlID>	4.1 SP3
Getting the Selection of an Input Control [page 546]	GET	/documents/<documentID>/reports/<reportID>/inputcontrols/<inputControlID>/selection	4.1 SP3
Setting the Selection of an Input Control [page 547]	PUT	/documents/<documentID>/reports/<reportID>/inputcontrols/<inputControlID>/selection	4.1 SP3
Deleting the Selection of an Input Control [page 548]	DELETE	/documents/<documentID>/reports/<reportID>/inputcontrols/<inputControlID>/selection	4.1 SP3
Input Controls and Lists of Values [page 526]	GET	/documents/<documentID>/reports/<reportID>/inputcontrols/<inputControlID>/lov	4.2 SP04
Input Controls and Lists of Values [page 526]	PUT	/documents/<documentID>/reports/<reportID>/inputcontrols/<inputControlID>/lov	4.2 SP04

2.10.3 Report Drill APIs

The REST APIs to drill through reports in a Web Intelligence document (`/documents/.../reports/.../driller/`).

The table below indicates the 4.0 and/or 4.1 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<reportID>`: report identifier retrieved from the list of reports

Action	Method	URL	Since
Getting the Drill Mode [page 377]	GET	/documents/<documentID>/reports/<reportID>/driller	4.0 SP6 | 4.1

Action	Method	URL	Since
Enabling the Query Drill [page 379]	POST	/documents/<documentID>/reports/<reportID>/driller	4.0 SP6 | 4.1
Updating the Drill Mode [page 377]	PUT	/documents/<documentID>/reports/<reportID>/driller	4.0 SP6 | 4.1
Disabling the Query Drill [page 380]	DELETE	/documents/<documentID>/reports/<reportID>/driller	4.0 SP6 | 4.1
Getting the Free Drill Elements [page 382]	GET	/documents/<documentID>/reports/<reportID>/driller/drillelements	4.0 SP6 | 4.1
Getting Information on the Drill Hierarchies [page 380]	GET	/documents/<documentID>/reports/<reportID>/driller/hierarchies	4.0 SP6 | 4.1
Performing a Drill [page 388]	POST	/documents/<documentID>/reports/<reportID>/driller/instructions	4.0 SP6 | 4.1
Making a Snapshot of a Report in Drill Mode [page 390]	POST	/documents/<documentID>/reports/<reportID>/driller/snapshot	4.0 SP6 | 4.1

2.10.4 Report Drill Filter APIs

The REST APIs to manage filters for drill through reports in a Web Intelligence document (`/documents/.../reports/.../driller/filters`).

The table below indicates the 4.0 and/or 4.1 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<reportID>`: report identifier retrieved from the list of reports
- `<filterID>`: drill filter identifier retrieved from the list of the drill filters

Action	Method	URL	Since
Creating a Drill Filter [page 383]	POST	/documents/<documentID>/reports/<reportID>/driller/filters	4.0 SP6 | 4.1
Getting the Drill Filters of a Report [page 384]	GET	/documents/<documentID>/reports/<reportID>/driller/filters	4.0 SP6 | 4.1
Getting the Details of a Drill Filter [page 385]	GET	/documents/<documentID>/reports/<reportID>/driller/filters/<filterID>	4.0 SP6 | 4.1
Updating a Drill Filter [page 386]	PUT	/documents/<documentID>/reports/<reportID>/driller/filters/<filterID>	4.0 SP6 | 4.1

Action	Method	URL	Since
Removing a Drill Filter [page 387]	DELETE	/documents/<documentID>/reports/<reportID>/driller/filters/<filterID>	4.0 SP6 | 4.1

2.11 Report Element APIs

The REST APIs to work with the elements of a report (/documents/.../reports/.../elements/). Use them to create, export a report element, or retrieve its dataset.

The table below indicates the 4.0 and/or 4.1 release in which support for the API was introduced.

Base URL: http://<server_name>:6405/biprws/raylight/v1

Path parameters:

- <documentID>: document identifier retrieved from the list of documents
- <reportID>: report identifier retrieved from the list of reports
- <elementID>: table identifier retrieved from the list of elements of a report

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Creating a Report Element [page 391]	POST	/documents/<documentID>/reports/<reportID>/elements	4.1 SP2
Getting the List of Report Elements [page 396]	GET	/documents/<documentID>/reports/<reportID>/elements	4.0 SP6 | 4.1
Getting the Details of a Report Element [page 400]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>	4.0 SP6 | 4.1
Exporting a Report Element [page 432]			
Inserting a Row or Column into a Table [page 429]	POST	/documents/<documentID>/reports/<reportID>/elements/<elementID>?strip=<strip>&position=<position>	4.1 SP6
Merging or Splitting Cells of a Table [page 430]	PUT	/documents/<documentID>/reports/<reportID>/elements/<elementID>?operation=<operation>&targetCellIds=<CellID[,...]>	4.1 SP6
Getting the Background of a Report Element [page 411]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>/background	4.1 SP5
Getting the Datapaths of a Report Element [page 412]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>/datapaths	4.1 SP3

Action	Method	URL	Since
Getting the Dataset of a Report Element [page 416]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>/dataset	4.1 SP3
Updating a Report Element [page 423]	PUT	/documents/<documentID>/reports/<reportID>/elements/<elementID>	4.1 SP2
Deleting a Report Element [page 438]	DELETE	/documents/<documentID>/reports/<reportID>/elements/<elementID>	4.1 SP2
Getting the Ranking in a Report Element [page 467]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>/ranking	

2.11.1 Report Element Axis APIs

The REST APIs to manage axes of report elements of type section, table, or chart (/documents/.../reports/.../elements/.../axes/).

The tables below indicate the 4.0 and/or 4.1 release in which support for the API was introduced.

Base URL: http://<server_name>:6405/biprws/raylight/v1

Path parameters:

- <documentID>: document identifier retrieved from the list of documents
- <reportID>: report identifier retrieved from the list of reports
- <elementID>: table identifier retrieved from the list of elements of a report
- <axisID>: axis identifier when the report element is a section, table, or chart
- <breakID>: break identifier retrieved from the details of the report element

Query parameters: see the detailed description of the related API.

Managing the Report Element Axis

Action	Method	URL	Since
Getting the Details of an Axis in a Report Element [page 441]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>	4.2 SP4
Updating the Axis of a Report Element [page 443]	PUT	/documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>	4.2 SP4

Expressions of a Report Element Axis

Action	Method	URL	Since
Updating the Expressions of an Axis [page 446]	PUT	/documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/expressions	4.1 SP3

Action	Method	URL	Since
Removing the Expressions From an Axis [page 447]	DELETE	/documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/expressions	4.1 SP3
Getting the Axis Expressions of a Report Element [page 445]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/expressions	4.2 SP4

Table Axes and Breaks

Action	Method	URL	Since
Updating the sortings of an axis. DEPRECATED. See Updating a Sort in a Report Element [page 489]	-	DEPRECATED from 4.2 SP04	4.1 SP3
Removing the sortings from an axis. DEPRECATED Removing a specific sort of a report element [page 492]	-	DEPRECATED from 4.2 SP04	4.1 SP3
Getting the Axis Sorts in a Report Element [page 460]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/sorts	4.1 SP4
Creating a Table Break [page 448]	POST	/documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/breaks	4.1 SP3
Updating a Table Break [page 450]	PUT	/documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/breaks/<breakID>	4.1 SP3
Deleting a Table Break [page 451]	DELETE	/documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/breaks/<breakID>	4.1 SP3
Listing all the Breaks on a Table's Axis [page 454]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/breaks	4.2 SP4
Getting the Details of a Break on a Table's Axis [page 462]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/breaks/<breakID>	4.2 SP4

2.11.2 Report Element Custom Property APIs

The REST APIs to work with custom properties of report elements (`/documents/.../reports/.../elements/.../properties`).

The table below indicates the 4.0 and/or 4.1 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<reportID>`: report identifier retrieved from the list of reports
- `<elementID>`: table identifier retrieved from the list of elements of a report
- `<propertyKey>`: key of the custom property of the report element

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Getting the List of Custom Properties [page 473]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>/properties	4.1 SP3
Getting the Custom Property Value [page 474]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>/properties/<propertyKey>	4.1 SP3
Adding a Custom Property to a Report Element [page 475]	POST	/documents/<documentID>/reports/<reportID>/elements/<elementID>/properties	4.1 SP3
Updating a Custom Property [page 476]	PUT	/documents/<documentID>/reports/<reportID>/elements/<elementID>/properties	4.1 SP3
Deleting a Custom Property [page 477]	DELETE	/documents/<documentID>/reports/<reportID>/elements/<elementID>/properties/<propertyKey>	4.1 SP3

2.11.3 Report Element Data Filter APIs

The REST APIs to work with the data filters that apply to report elements (`/documents/.../reports/.../elements/.../datafilter`).

The table below indicates the 4.0 and/or 4.1 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<reportID>`: report identifier retrieved from the list of reports
- `<elementID>`: table identifier retrieved from the list of elements of a report

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Creating a Data Filter for a Report Element [page 478]	POST	/documents/<documentID>/reports/<reportID>/elements/<elementID>/datafilter	4.1 SP2

Action	Method	URL	Since
Getting the Details of a Data Filter for a Report Element [page 480]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>/datafilter	4.1 SP2
Updating the Data Filter for a Report Element [page 481]	PUT	/documents/<documentID>/reports/<reportID>/elements/<elementID>/datafilter	4.1 SP2
Deleting a Data Filter from a Report Element [page 483]	DELETE	/documents/<documentID>/reports/<reportID>/elements/<elementID>/datafilter	4.1 SP2

2.11.4 Report Element Calculation APIs

The REST APIs to manage calculations in report elements (/documents/.../reports/.../elements/.../calculations/).

The table below indicates the release in which support for the API was released.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<reportID>`: report identifier retrieved from the list of reports
- `<elementID>`: table identifier retrieved from the list of elements of a report
- `<calculationID>`: calculation identifier retrieved from the list of calculations of a report

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Getting the Calculations of a Report Element [page 468]	GET	/documents/<documentId>/reports/<reportId>/elements/<elementId>/calculations	4.3
Getting the Details of a Calculation [page 470]	GET	/documents/<documentId>/reports/<reportId>/elements/<elementId>/calculations/<calculationId>	4.3
Creating a Calculation [page 471]	POST	/documents/<documentId>/reports/<reportId>/elements/<elementId>/calculations	4.3

Action	Method	URL	Since
Deleting a Calculation [page 472]	DELETE	/documents/<documentId>/reports/<reportId>/elements/<elementId>/calculations/<calculationId>	4.3

2.11.5 Report Element Ranking APIs

The REST APIs to work with rankings in report elements of type table, section, chart, or custom element (/documents/.../reports/.../elements/.../ranking).

The table below indicates the 4.0 and/or 4.1 release in which support for the API was introduced.

Base URL: http://<server_name>:6405/biprws/raylight/v1

Path parameters:

- <documentID>: document identifier retrieved from the list of documents
- <reportID>: report identifier retrieved from the list of reports
- <elementID>: table identifier retrieved from the list of elements of a report

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Creating a Ranking in a Report Element [page 464]	POST	/documents/<documentID>/reports/<reportID>/elements/<elementID>/ranking	4.1 SP3
Updating the Ranking of a Report Element [page 465]	PUT	/documents/<documentID>/reports/<reportID>/elements/<elementID>/ranking	4.1 SP3
Deleting the Ranking of a Report Element [page 466]	DELETE	/documents/<documentID>/reports/<reportID>/elements/<elementID>/ranking	4.1 SP3
Getting the Ranking in a Report Element [page 467]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>/ranking	

2.11.6 Report Element Sorting APIs

The REST APIs to manage sorts of report elements of type section, table, or chart (/documents/.../reports/.../elements/.../sortID).

The tables below gives an overview of the Sorting APIs provided with the 4.2 SP04 release..

Base URL: http://<server_name>:6405/biprws/raylight/v1

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<reportID>`: report identifier retrieved from the list of reports
- `<elementID>`: table identifier retrieved from the list of elements of a report
- `<sortID>`: sort identifier

Managing Sorts in a Report Element

Action	Method	URL	Since
Getting sorts in a report element [page 484]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>/sorts	4.2 SP4
Updating the sorts of a report element [page 487]	PUT	/documents/<documentID>/reports/<reportID>/elements/<elementID>/sorts	4.2 SP4
Removing the Sorts of a Report Element [page 490]	DELETE	/documents/<documentID>/reports/<reportID>/elements/<elementID>/sorts/	4.2 SP4
Getting the Details of a Sort [page 486]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>/sorts/<sortID>	4.2 SP4
Updating a Sort in a Report Element [page 489]	PUT	/documents/<documentID>/reports/<reportID>/elements/<elementID>/sorts/<sortID>	4.2 SP4
Removing a specific sort of a report element [page 492]	DELETE	/documents/<documentID>/reports/<reportID>/elements/<elementID>/sorts/<sortID>	4.2 SP4

2.12 Data Provider APIs

The REST APIs to work with queries to data sources in documents (`/documents/.../dataproviders/`). Data providers can be universes, BEx queries, Excel spreadsheets and free-hand SQL scripts.

The table below indicates the release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<dataProviderID>`: identifier of the data provider available for a document
- `<flowID>`: flow identifier that you can know from the flow count

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Getting the List of Data Providers [page 555]	GET	/documents/<documentID>/dataproviders	4.0 SP5 | 4.1

Action	Method	URL	Since
Getting the Details of a Data Provider [page 557]	GET	/documents/<documentID>/dataproviders/<dataProviderID>	4.0 SP5 | 4.1
Adding a Data Provider [page 560]	POST	/documents/<documentID>/dataproviders	4.0 SP6 | 4.1
Moving a Data Provider in a Document [page 571]	PUT	/documents/<documentID>/dataproviders?fromId=<DPn>&toId=<DPx>	4.0 SP6 | 4.1
Updating a Data Provider [page 563]	PUT	/documents/<documentID>/dataproviders/<dataProviderID>	4.0 SP5 | 4.1
Deleting a Data Provider [page 572]	DELETE	/documents/<documentID>/dataproviders/<dataProviderID>	4.0 SP6 | 4.1
Getting the Flow Count of a Data Provider [page 583]	GET	/documents/<documentID>/dataproviders/<dataProviderID>/flows/count	4.0 SP5 | 4.1
Getting the Details of a Flow [page 584]	GET	/documents/<documentID>/dataproviders/<dataProviderID>/flows/<flowID>	4.0 SP5 | 4.1
Getting the Samples of a Flow [page 586]	GET	/documents/<documentID>/dataproviders/<dataProviderID>/flows/<flowID>/samples	4.1 SP5
Getting the Query Plan [page 587]	GET	/documents/<documentID>/dataproviders/<dataProviderID>/queryplan	4.1 SP4
Updating the Query Plan [page 590]	PUT	/documents/<documentID>/dataproviders/<dataProviderID>/queryplan	4.1 SP4
Getting the Query Specification [page 592]	GET	/documents/<documentID>/dataproviders/<dataProviderID>/specification	4.0 SP6 | 4.1
Updating the Query Specification [page 593]	PUT	/documents/<documentID>/dataproviders/<dataProviderID>/specification	4.0 SP6 | 4.1

2.12.1 Data Provider Change Source APIs

Use these APIs to replace a data provider with another in a document (`/documents/.../dataproviders/.../mappings/`).

The tables below indicate the 4.0 and/or 4.1 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<dataProviderID>`: identifier of the data provider available for a document

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Getting the Possible Object Mappings Using the Default Strategies [page 573]	GET	/documents/<documentID>/dataproviders/<dataProviderID>/mappings?originDataproviderIds=<DP1ID>,<DP2ID>&targetDatasourceId=<DatasourceID>	4.0 SP6 | 4.1
Getting the Possible Object Mappings Using Selected Strategies [page 576]	PUT	/documents/<documentID>/dataproviders/<dataProviderID>/mappings?originDataproviderIds=<DP1ID>,<DP2ID>&targetDatasourceId=<DatasourceID>	4.1 SP6
Changing the Data Objects of a Data Provider [page 579]	POST	/documents/<documentID>/dataproviders/<dataProviderID>/mappings?originDataproviderIds=<DP1ID>,<DP2ID>&targetDatasourceId=<DatasourceID>	4.0 SP6 | 4.1

2.12.2 Data Provider Refresh APIs

Use these APIs to refresh parameters from data providers (`/documents/.../dataproviders/.../parameters`).

The table below indicates the 4.0 and/or 4.1 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<dataProviderID>`: identifier of the data provider available for a document

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Getting the Refresh Parameters of a Data Provider [page 665]	GET	/documents/<documentID>/dataproviders/<dataProviderID>/parameters	4.1 SP5
Refreshing a Data Provider [page 666]	PUT	/documents/<documentID>/dataproviders/<dataProviderID>/parameters	4.1 SP5

2.13 Publication API

The rest API to work with publications.

The table below indicates the release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<publicationID>`: publication identifier retrieved from the list of publications.

Action	Method	URL	Since
Getting the Details of a Publication [page 677]	`GET`	`/publications/<publicationID>`	4.3

2.14 Variant APIs

The REST APIs to manage variants in Web Intelligence documents (`/documents/.../variants/`).

The table below indicates the 4.2 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<variantID>`: variant identifier retrieved from the list of variants of the document

Query parameters: see the detailed description of the related API.

Variants

Action	Method	URL	Since
Getting the List of Variants [page 654]	`GET`	`/documents/<documentID>/variants`	4.2 SP3
Getting the Details of a Variant [page 655]	`GET`	`/documents/<documentID>/variants/<variantID>`	4.2 SP3
Creating a Variant [page 660]	`POST`	`/documents/<documentID>/variants`	4.2 SP3
Editing a Variant [page 662]	`PUT`	`/documents/<documentID>/variants/<variantID>`	4.2 SP3
Deleting a Variant [page 663]	`DELETE`	`/documents/<documentID>/variants/<variantID>`	4.2 SP3

2.15 SAP BW Connection and BEx Query APIs

The REST APIs to manage SAP BW connections and BEx queries (`/bwconnections`).

The table below indicates the 4.0 and/or 4.1 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameter: `<bwConnectionID>` is the connection identifier retrieved from the list of SAP BW connections.

Action	Method	URL	Since
Getting the List of SAP BW Connections [page 607]	GET	/bwconnections	4.0 SP6 | 4.1
Getting the Details of an SAP BW Connection [page 609]	GET	/bwconnections/<bwConnectionID>	4.0 SP6 | 4.1
Browsing the Details of an SAP BW connection [page 610]	PUT	/bwconnections/<bwConnectionID>	4.0 SP6 | 4.1
Getting the Outline of a BEx Query [page 612]	PUT	/bwconnections/<bwConnectionID>/outline	4.0 SP6 | 4.1
Getting the Capabilities of a BEx Query [page 614]	PUT	/bwconnections/<bwConnectionID>/capabilities	4.0 SP6 | 4.1

2.16 Search Resource API

The REST API to search for resources like universes, connections, and folders on the CMS repository (/searches).

The table below indicates the 4.0 and/or 4.1 release in which support for the API was introduced.

Base URL: http://<server_name>:6405/biprws/raylight/v1

Action	Method	URL	Since
Getting Resources [page 691]	POST	/searches	4.1 SP6

2.17 Comments APIs

The REST APIs to get comments (/comments).

The table below indicates the 4.2 release in which support for the API was introduced.

Base URL: http://<server_name>:6405/biprws/raylight/v1

Path parameters:

- <documentID>: document identifier retrieved from the list of documents
- <reportID>: report identifier retrieved from the list of reports
- <pageRange>: page numbers in report.
- <elementId>: identifier of an element in a report
- <commentId>: identifier of a comment

Query parameters: see the detailed description of the related API.

Action	Method	URL	Since
Getting the Visible Comments of a Report [page 373]	GET	/documents/<{documentId}>/reports/<{reportId}>/pages/comments	4.2 SP04
Getting the Visible Comments of a Page Range of a Report [page 374]	GET	/documents/<{documentId}>/reports/<{reportId}>/ pages/<{pageRange}>/comments	4.2 SP04
Getting all the Comments or Only the Visible Comments of a Report Element [page 493]	GET	/documents/<{documentId}>/reports/<{reportId}>/elements/<{elementId} >/comments	4.2 SP04
Getting a Specific Comment of a Report Element [page 495]	GET	/documents/<{documentId}>/reports/<{reportId}>/elements/<{elementId} >/comments/<{commentId}>	4.2 SP04
Adding a New Comment to a Report Element [page 496]	POST	/documents/<{documentId}>/reports/<{reportId}>/elements/<{elementId}>/comments	4.2 SP05
Modfiying a Specific Message in a Comment Thread [page 497]	PUT	documents/<{documentId}>/reports/<{reportId}>/elements/<{elementId}>/comments/<{commentId}>	4.2 SP05
Deleting a Message in a Comment Thread [page 498]	DELETE	/documents/<{docId}>/reports/<{repId}>/elements/<{elId}>/comments/<{commentId}>	4.2 SP05

2.18 Session Security API

Use this API to return the security rights of the current session.

Get the allowed security rights of the connected user for Web Intelligence application.

Action	Method	URL	Since
Getting the Web Intelligence Security Rights for the Current User [page 163]	GET	/session/rights	4.2 SP4

Related Information

Getting a Document's Security Rights for the Current User [page 231]

2.19 Web Intelligence OData APIs

The REST APIs to retrieve Web Intelligence documents content through OData protocol.

The table below indicates the 4.3 release in which support for the API was introduced.

Base URL: `http://<server_name>:<port>/biprws/raylight/v1`

Path parameters:

- `<documentCUID>` is the document identifier (CUID) retrieved from the list of documents.
- `<reportID>` is the report identifier containing the report element to retrieve.
- `<elementID>` is the identifier of the targeted report element.
- `<dataProviderID>` is a data provider identifier in a document.
- `<flowID>` is the identifier of a flow in a query.
- `<scheduleCUID>` is a recurring instance identifier (CUID) of the document `<documentCUID>`.

Action	Method	URL	Since
Getting a Document's Cubes and Report Elements [page 703]	GET	`documents/cuid_<documentCUID>/datamodel/data.svc`	4.3 SP3
Getting a Document's Metadata [page 703]	GET	`documents/cuid_<documentCUID>/datamodel/data.svc/$metadata`	4.3 SP3
Getting a Document's Cube Dataset [page 704]	GET	`documents/cuid_<documentCUID>/datamodel/data.svc/cube_<dataProviderID>-<flowID>`	4.3 SP3
Getting a Document's Report Element Dataset [page 704]	GET	`documents/cuid_<documentCUID>/datamodel/data.svc/re_<reportID>-<elementID>`	4.3 SP3
Getting a Schedule's Last Instance's Cubes and Report Elements [page 705]	GET	`documents/cuid_<documentCUID>/schedules/cuid_<scheduleCUID>/datamodel/data.svc`	4.3 SP3
Getting a Schedule's Last Instance Metadata [page 705]	GET	`documents/cuid_<documentCUID>/schedules/cuid_<scheduleCUID>/datamodel/data.svc/$metadata`	4.3 SP3
Getting a Schedule's Last Instance's Cube Dataset [page 706]	GET	`documents/cuid_<documentCUID>/schedules/cuid_<scheduleCUID>/datamodel/data.svc/cube_<dataProviderID>-<flowID>`	4.3 SP3
Getting a Schedule's Last Instance's Report Element Dataset [page 706]	GET	`documents/cuid_<documentCUID>/schedules/cuid_<scheduleCUID>/datamodel/data.svc/re_<reportID>-<elementID>`	4.3 SP3
Getting a Report Element Data Flow Name (Deprecated) [page 707]	GET	`documents/<documentCUID>/reports/<reportID>/elements/<elementID>/data.svc/`	4.3 SP2
Getting a Report Element Metadata (Deprecated) [page 708]	GET	`documents/<documentCUID>/reports/<reportID>/elements/<elementID>/data.svc/$metadata`	4.3 SP2

Action	Method	URL	Since
Getting a Report Element Data (Deprecated) [page 708]	GET	`documents/<documentCUID>/reports/<reportID>/elements/<elementID>/data.svc/Flows0`	4.3 SP2

3 Introduction to the RESTful Web Service SDK User Guide

This guide relates to the SAP BusinessObjects Business Intelligence platform 4.2 Support Package 3 release.

What this Guide is About

The *SAP BusinessObjects RESTful Web Service SDK User Guide for Web Intelligence and the BI Semantic Layer* provides conceptual and reference information on the BI Semantic Layer RESTful Web Service SDK and the Web Intelligence RESTful Web Service SDK:

- Purpose and concepts
- How to use the samples provided with the SDKs
- HTTP methods and URLs of the REST APIs
- Request and response examples of the REST APIs

Audience [page 45]

Conventions in This Guide [page 45]

Key Tasks [page 46]

Use Cases [page 46]

3.1 Audience

The SAP BusinessObjects RESTful Web Service SDK User Guide for Web Intelligence and the BI Semantic Layer is intended for developers responsible for writing programs that access and consume the BI platform RESTful web services. These programs can be embedded within Business Intelligence solutions on the BI platform.

3.2 Conventions in This Guide

For consistency and readability across this guide:

- Request and response body schemas are described in XML format without closing tags. On the contrary, in the REST API call examples, requests and responses are described thoroughly in either XML or JSON format.
- The REST API references do not specify the default base URL, which is described in the *Default Base URLs* section.

3.3 Key Tasks

This guide provides you with key information for using the RESTful Web Service SDKs. For each of the following tasks, see the appropriate section:

Task	Description
To retrieve the universe description	Getting the Details of a Universe (Semantic Layer) [page 168]
To design a query specification	Queries [page 105]
To query a universe	Creating a Query [page 180]
To get the query results	Managing Query Results [page 205]
To create an empty Web Intelligence document	Creating a Document [page 224]
To add a data source to a document	Adding a Data Provider [page 560]
To add a query specification to a document	Updating the Query Specification [page 593]
To get document data	Refreshing a Document [page 626]
To create an empty report	Creating a Report [page 339]
To create elements in a report	Creating a Report Element [page 391]
To format a report	Updating a Report Element [page 423]
To refresh a document	Refreshing Documents [page 619]
To schedule a document	Adding a Schedule [page 671]

3.4 Use Cases

The SAP BusinessObjects RESTful Web Service SDKs allow you to access the resources of the SAP BusinessObjects BI platform (universes, Web Intelligence documents and reports) stored in the CMS repository using the HTTP protocol. You implement CRUD (Create, Read, Update, Delete) operations so that the provided web services send requests over HTTP and receives responses in the XML or JSON format.

For example, you can use these SDKs to retrieve data from a data provider and expose it through reports built with report elements. You can perform report scheduling and drill through a report to explore and analyze business data. You can also browse universe metadata, submit and execute queries.

The SDKs allow you to address some business requirements that can include:

- Exposing universe metadata within a custom application
- Building a custom query panel and embedding it anywhere
- Creating queries on the fly during analysis
- Running queries directly into custom scripts, Java applications, and so on
- Running queries to feed a client tool with business data
- Embedding documents and reports in non-SAP client tools or web applications

- Working with documents and reports within custom applications

4 Using the RESTful Web Service SDKs

REST Methodology

The SDKs comply with the REST methodology. You can access them using any programming language that supports making HTTP requests. You can send HTTP requests from any operating system, either Microsoft Windows or UNIX, or even from a mobile platform. You can also make HTTP requests without writing code by using tools that make HTTP requests.

Before You Begin

1. Make sure the WebApplicationContainerServer (WACS) server is running. This server hosts the RESTful web services.
2. Decide to trace the server activity.
3. Retrieve the base URL. See Default Base URLs [page 50].
4. Log on to the BI platform and get the logon token. See To Log on to the BI platform [page 51].

See the *Business Intelligence Platform Administrator Guide* to learn about:

- Starting the WACS server
- Activating trace log
- Displaying the trace

See the *BI Platform RESTful Web Service Developer Guide* to learn about:

- Setting up the development environment
- Installing the RESTful web services for a custom installation
- Retrieving the base URL, either from the CMC or with a program
- Logging on to the BI platform RESTful Web Service SDK
- Logging off from the BI platform RESTful Web Service SDK

The RESTful Web Service SDKs also rely on the BI platform RESTful Web Service API for session management and repository access.

Format

Both XML and JavaScript Object Notation (JSON) request and response formats are supported. For more information on the JSON format, see www.json.org and RFC4627 . For more information on the XML format, see www.w3.org/XML .

The SDKs require the following JSON syntax in request bodies:

- The at sign (@) for attributes
- The dollar symbol ($) for element values

For example:

XML	JSON

`<schedule>`
`  <id>9439</id>`
`  <name>now-schedule</name>`
`  <format type="webi"/>`
`  <status id="1">Completed</status>`
`  <destination>`
`  <ftp>`
`  <host>vs0202</host>`
`  <port>21</port>`
`  <username>admin</ username>`
`  <account/>`
`  <directory>./</directory>`
`  </ftp>`
`  </destination>`
`</schedule>`
	`{`
	`  "schedule": {`
	`  "id": "9439",`
	`  "name": "now-schedule",`
	`  "format": {"@type": "webi"},`
	`  "status": {`
	`  "@id": 1,`
	`  "$": "Completed"`
	`  },`
	`  "destination": {`
	`  "ftp": {`
	`  "host": "vs0202",`
	`  "port": "21",`
	`  "username": "admin",`
	`  "directory": "./"`
	`  }`
	`  }`
	`  }`
	`}`

The JSON format supported in response bodies has changed in the 4.2 SP3 release. See SAP note n° 2312733 "Web Intelligence RESTful Web Service does not return valid empty arrays and singletons in JSON".

	Before 4.2 SP3	Since 4.2 SP3
If a JSON array has no child	`{"reports": {"report":""}}`	`{"reports":{"report": []}}`

If a JSON array has one child	`{"reports":{`
	`  "report":`
	`  {…}`
	`  }`
	`}`
		`{"reports":{`
		`  "report":[`
		`  {…}`
		`  ]`
		`  }`
		`}`

If a JSON array has children - no change	`{"reports":{"report": [{…},{…},{…}]}}`	`{"reports":{"report": [{…},{…},{…}]}}`

Default Base URLs [page 50]

To Log on to the BI platform [page 51]

Document State [page 51]

Although Web Intelligence follows the REST Web Service principles, document management isn't completely stateless for performance reasons.

Getting Web Intelligence RESTful Web Service SDK Information [page 52]

Getting the Logged-in User Details [page 54]
Supporting Multiple Languages [page 55]
Dispensable Information in a Request Body [page 55]
HTTP Request Status [page 57]
Success and Error Messages [page 58]

4.1 Default Base URLs

To use the RESTful web services for Web Intelligence and the BI Semantic Layer, you must know the protocol, server name, port number and path of the service that listens to the HTTP requests. You configure the default base URL in the CMC from *Applications* > *REST Web Service* > *Properties* > *Access URL*. See chapter 12 of the *Business Intelligence Platform Administrator Guide* for more information.

Basic installations of the BI platform that are installed on a single server use the default base URLs:

Web Service SDK	URL
BI platform	`http://<server_name>:6405/biprws`
BI Semantic Layer	`http://<server_name>:6405/biprws/sl/v1`
Web Intelligence	`http://<server_name>:6405/biprws/raylight/v1`

> **Note**
> 6405 is the default HTTP port number used by the RESTful web services.

Conventions in this Guide

The following conventions apply to the REST APIs described in this guide:

- `<base_bip_REST_URL>` stands for `http://<server_name>:6405/biprws`.
- `<base_webi_REST_URL>` stands for `http://<server_name>:6405/biprws/raylight/v1`.
- `<base_sl_REST_URL>` stands for `http://<server_name>:6405/biprws/sl/v1`.

> **Note**
> The API references do not show the default base URLs to make the reading easier.

4.2 To Log on to the BI platform

You need a logon token to authenticate to the BI platform. The following procedure describes the common way to get this token.

1. Get the authentication information.

   Request:

   ```
   GET http://<server_name>:6405/biprws/logon/long
   ```

   Response:

   ```
   <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>
   ```

2. Get the logon token by using the response of step 1 as request body with your BI platform credentials.

   Request:

   ```
   POST http://<server_name>:6405/biprws/logon/long
   ```

   Request body filled with BI platform credentials:

   ```
   <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>
   ```

   Response:

   ```
   <attrs xmlns="http://www.sap.com/rws/bip">
       <attr name="LogonToken" type="string">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=SFY6agrLPxpfQBK1ZKYCwoBZKCbfsQm7Vg
   WZFiH.RhM,UP</attr>
   </attrs>
   ```

Add the logon token value between double quotes to the `X-SAP-LogonToken` request header of any further request.

4.3 Document State

Although Web Intelligence follows the REST Web Service principles, document management isn't completely stateless for performance reasons.

When creating or editing a document, it is not immediately saved in the BI platform repository.

Changes are kept in memory until you save the document using the PUT request. A state is maintained to indicate whether changes are still ongoing on the document.

To actually save changes, you can use the `POST /documents/<documentId>` or `PUT /documents/<documentId>` calls.

Related Information

Saving a Document [page 233]
Updating the State of a Document [page 247]

4.4 Getting Web Intelligence RESTful Web Service SDK Information

Usage

Returns the following information about your running version of SAP BusinessObjects Web Intelligence RESTful Web Service SDK:

- Product name
- Version number
- Vendor name
- Major release, minor release, Support Package, patch, and build numbers
- Timestamp
- Copyright

> **Note**
> - The result is computed at build time, not running time, and depends if you are connected or not.
> - You can access this URL without authentication in any Web browser.

Request

```
GET /about
```

Response

Response type: `application/xml` or `application/json`

Example

XML

```
<about>
    <title>Web Intelligence - Raylight</title>
    <version>1.0</version>
    <vendor>SAP SE. All rights reserved.</vendor>
    <build major="14" minor="1" sp="5" patch="8" number="1754">14.1.5.1754</
build>
    <timestamp>20150805.031625</timestamp>
    <copyright>?©2010 - 2017 SAP SE or an SAP affiliate company. All rights
reserved.</copyright>
</about>
```

JSON

```
{"about":
    {"title":"Web Intelligence - Raylight",
     "version":1,
     "vendor":"SAP SE",
     "build": {
         "@major": "14",
         "@minor": "1",
         "@sp": "5",
         "@patch": "8",
         "@number": "1754",
         "$": "14.1.5.1754"
         },
     "timestamp":"20150805.031625",
     "copyright":"?©2010 - 2017 SAP SE or an SAP affiliate company. All rights
reserved."
    }
}
```

Example: With authentication

With authentication, the result is computed at run time and comes mostly from the BOE server. Since a session has been established, localization is also possible.

`GET /about`

Accept: `application/xml`

XML

```
<about>
       <title>Web Intelligence</title>
       <version>1.0</version>
       <vendor>© 2010-2017 SAP SE. All rights reserved.</vendor>
       <build major="14" minor="2" sp="4" patch="0" number="912">14.2.4.912</
build>
       <timestamp>20170314.000118</timestamp>
       <copyright>© 2010-2017 SAP SE or an SAP affiliate company. All rights
reserved. SAP and other SAP products and services mentioned herein as well
as their respective logos are trademarks or registered trademarks of SAP SE
in Germany and other countries. Please see http://www.sap.com/corporate-en/legal/
copyright/index.epx#trademark for additional trademark information and notices.</
copyright>
```

```
</about>
```

4.5 Getting the Logged-in User Details

Usage

Returns the following information about the logged-in user:

- Identifier
- User name
- CUID number
- Timezone
- Preferred Viewing Locale
- Product Locale

Request

```
GET /session
```

Response

Response type: `application/xml` or `application/json`

Example

XML

Response:

```
<session>
    <user>
        <id>6357</id>
        <name>user</name>
        <cuid>AZFmWPGBwwRBm3almUJGkfk</cuid>
        <timeZone>Japanische Normalzeit</timeZone>
        <preferredViewingLocale>ja-JP</preferredViewingLocale>
        <productLocale>de</productLocale>
    </user>
</session>
```

JSON

Response:

```
{
    "session": {
        "user": {
            "id": 6357,
            "name": "user",
            "cuid": "AZFmWPGBwwRBm3almUJGkfk"
            "timeZone": "Japanische Normalzeit",
            "preferredViewingLocale": "ja-JP",
            "productLocale": "de"
        }
    }
}
```

4.6 Supporting Multiple Languages

The RESTful Web Service SDKs allow you to work in multilingual environments. You can request Web Intelligence documents and system messages to be returned in your preferred language. Include the following declarations in the request header:

Header	Value
`Accept-Language`	The preferred language used to retrieve system and error messages. This corresponds to the Product Locale (PL) of the BI platform. To get system messages returned in French, set `Accept-Language` to `fr-FR`.
`X-SAP-PVL`	The preferred language used to retrieve BI resource content such as Web Intelligence documents. This corresponds to the Preferred Viewing Language (PVL). To request all documents available in German, set `X-SAP-PVL` to `de-DE`.

The RESTful web service opens one instance of the Web Intelligence document in the client machine memory for each Preferred Viewing Locale requested by the client.

4.7 Dispensable Information in a Request Body

You can send a `POST` or `PUT` HTTP request with a body to create or update a Web Intelligence resource. If you provide information that is not needed to perform the requested action, the method simply ignores it and returns a successful message.

Example

You want to update some `<dataLabels>` property of a Waterfall chart by sending the following request:

```
PUT <base_webi_REST_URL>/documents/8084/reports/12/elements/27
```

In SAP BusinessObjects Web Intelligence, a Waterfall chart does not contain any legend. However, the request body you are sending contains a legend as follows:

```
<element type="Visualization">
    <parentId>2</parentId>
    <content>
        <chart type="Waterfall">
            <layout showDimensionsWithEmptyMeasureValues="true"
showDimensionsWithMeasuresEqualToZero="true"
                showDimensionsWithSumOfMeasuresEqualToZero="true"
showMeasuresWithEmptyDimensionValues="false"
                showTotal="true" parentAsTotal="false"
duplicateRowAggregation="true" horizontal="false"/>
            <title visible="true">
                ...
            </title>
            <legend visible="true">
                <style>
                    <border thickness="None">
                        <color rgb="#000000" alpha="0"/>
                    </border>
                    <background>
                        <color rgb="#000000" alpha="0"/>
                    </background>
                    <font size="12" face="Helvetica" italic="true" bold="true"
strikethrough="false"
                        underline="false" rgb="#00c7ff"/>
                    <alignment horizontal="Left" vertical="Center"
textPolicy="Wrap"/>
                </style>
                ...
            </legend>
            <dataLabels type="Value" visible="false">
                <style>
                    <border thickness="None">
                        <color alpha="255" rgb="#000000"/>
                    </border>
                    <background>
                        <color alpha="0" rgb="#000000"/>
                    </background>
                    <font size="10" face="Colibri" italic="false" bold="true"
strikethrough="false"
                        underline="false" rgb="#ffffff"/>
                </style>
            </dataLabels>
        </chart>
    </content>
</element>
```

The REST Web Service SDK ignores the `<legend>` part and updates the chart as requested.

The response is the following message:

```
<success>
    <message>The resource of type "Report element" with identifier "27" has been
successfully updated.</message>
    <id>27</id>
</success>
```

4.8 HTTP Request Status

The response of an HTTP request contains a `Status Code` attribute that gives information on the success or error of the request call.

See the table below to learn about the codes that provide status for HTTP requests.

Code	Status	Description
200	Success	Successful request.
400	Bad request	The requested resource exists, but the request contains errors.
401	Failed to logon or invalid session	Logon failed. Check that the user name, password, and server name are correct. The current session may have expired. Log on to obtain a new session.
403	Access denied	You do not have permission to operate on the requested resource.
404	Service is not available	The requested service is not provided by the RESTful Web Service 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	The web service is not available. Verify that the service is configured correctly.

Example: Content of a Response Header

This example shows a successful response code.

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Date: Fri, 15 Jun 2012 10:14:15 GMT
Content-Type: application/xml
Content-Length: 204
```

4.9 Success and Error Messages

Success Messages

If the request is successful, HTTP calls to REST APIs return a message in the XML or JSON format, as follows:

```
{"success":
    {"message": "a success message",
     "id": "an identifier"
    }
}
```

Where `id` refers to the identifier of the object concerned, such as a query, document, variable, or style. In some rare cases, `id` is not present.

Example: JSON Success Message

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Date: Tue, 13 Jan 2015 10:14:15 GMT
Content-Type: application/json
Content-Length: 204
{"success":
    {"message": "The resource of type \"Document\" with identifier \"16706\" has
been successfully updated.",
     "id": "16706"
    }
}
```

Error Messages

If the request was unsuccessful, the call returns a message in the XML or JSON format, as follows:

```
<error>
    <error_code>An error code</error_code>
    <message>An error message</message>
```

```
</error>
```

The following table shows the possible Web Intelligence RESTful Web Service SDK error codes. Each of these codes addresses a category of error messages.

Error Code	Description
WSR 00001	The user does not provide session token. The session is not found.
WSR 00002	The session token is invalid.
WSR 00100	A rule is not respected.
WSR 00101	An argument is not correct.
WSR 00102	The request body is malformed.
WSR 00103	The request is invalid.
WSR 00400	The user tries to get a resource that does not exist.
WSR 00401	The user tries to create a resource that already exist.
WSR 00402	The user fails to access a resource.
WSR 00501	The user tries to perform an action that is not supported.
WSR 00999	This is an internal error.

The BI Semantic Layer RESTful Web Service SDK error codes are more specific. Each of these codes corresponds to a particular error message and to a specific error. See the *Error Messages Explained* guide for a detailed description of the error messages.

You can display the full error stack traces for debugging purpose by checking the *Show Error Stack* property for your WACS server in the CMC. See the *Business Intelligence Platform Administrator Guide* for the complete procedure.

> **Note**
>
> The API reference chapters do not describe the call responses if an error occurs.

Example: 400 Error Code

```
Content-Type: application/xml
HTTP Response Code: 404
<?xml version="1.0" encoding="UTF-8"?>
<error>
    <error_code>WSR 00400</error_code>
    <message>The expression "[DUMMY]" cannot be found in the document
dictionary.</message>
</error>
```

Example: 401 Error Code

```
Content-Type: application/xml
HTTP Response Code: 400
<?xml version="1.0" encoding="UTF-8"?>
<error>
    <error_code>WSR 00401</error_code>
    <message>The resource of type "Report" with identifier "154769" already has
an inner resource of type "Data filter".</message>
</error>
```

5 Using the RESTful Web Service Samples

Code samples are provided to demonstrate the usage of the Web Intelligence and BI Semantic Layer RESTful Web Service SDKs.

The following samples are Java code snippets that allow you to understand how a Java application can be implemented to run HTTP calls to the REST APIs. You can use these samples as a basis to facilitate and accelerate your own Java programming.

Samples for the BI Semantic Layer REST APIs are supplied in the archive `C:\Program Files (x86)\SAP Business Objects\SAP BusinessObjects Enterprise XI 4.0\SL SDK\SDK Samples\SLRESTWebService.zip`.

Samples for the Web Intelligence REST APIs are supplied in the archive `C:\Program Files\SAP BusinessObjects\SAP BusinessObjects Enterprise XI 4.0\Samples\webi\RaylightRESTWS_Samples.zip`.

About the BI Semantic Layer REST API Samples [page 61]

About the Web Intelligence REST API Samples [page 63]

5.1 About the BI Semantic Layer REST API Samples

The SDK samples described in the following table work with the `SPL_Warehouse.unx` sample universe. You can install this universe on your system by selecting *Developer Tools* > *Warehouse Database and Universe Sample* from the server installation setup. Find information about this universe on the SAP Community Network at http://scn.sap.com/docs/DOC-22145.

Class	Description
`GetUniverseMetadataTest`	Returns the details of the universe using `GET /universes/<universeID>`
`GetUniversesTest`	Returns the list of the universes in the CMS repository using `GET /universes?offset&limit`
`GetUniverseViews`	Returns the views of the universe using `GET /universes/<universeID>/businessviews`
`QueryExecutionTest`	Creates a query on top of the universe: • Creates a query using `POST /queries` • Runs the query using `GET /queries/<queryID>/data.svc` • Returns the query results using `GET /queries/<queryID>/data.svc/<flowName>` • Deletes the query using `DELETE /queries/<queryID>`

Class	Description
`QueryParametersTest`	Creates a query on top of the universe and answers parameters this query requires: • Creates a query with parameters using `POST /queries` • Returns the query parameters using `GET /queries/<queryID>/parameters` • Runs the query using `GET /queries/<queryID>/data.svc` • Returns the query results using `GET /queries/<queryID>/data.svc/<flowName>` • Deletes the query using `DELETE /queries/<queryID>`

resources and util Packages

The `com.sap.sl.sdk.consumption.samples.resources` package contains XML descriptions used in sample requests, such as the query specification. The `com.sap.sl.sdk.consumption.samples.util` package provides classes that help to perform utility actions in the samples (for example, login, fill in CMS parameters, send a request, and so on).

To Deploy the Samples with Eclipse [page 62]

5.1.1 To Deploy the Samples with Eclipse

You can run Java samples in your Eclipse project.

1. Launch Eclipse by double-clicking the `eclipse.exe` file and select your workspace.

   Use Eclipse 3.6 or higher.
2. Create a project.
3. Select *File* > *Import*.
   The Import dialog box appears.
4. Expand *General*, select *Existing Projects into Workspace*, and click *Next*.
5. Select the *Select archive file* option and click *Browse*.
6. Browse to the sample directory and select `SLRESTWebService.zip`.
7. In the *Projects* area, select *com.sap.sl.sdk.consumption.samples* and click *Finish*.
   The `com.sap.sl.sdk.consumption.samples` project folder appears in the *Package Explorer* view.
8. Under *com.sap.sl.sdk.consumption.samples.util*, open the `BaseTest.java` file for editing.
9. Fill in the following parameters with the values corresponding to your installation:
   - `CMS_SERVER_URL`: the URL exposed by the server hosting the RESTful web services
   - `CMS_USER`: user name used in your samples to connect to the CMS repository
   - `CMS_PASSWORD`: password used in your samples to connect to the CMS repository
10. Under *com.sap.sl.sdk.consumption.samples*, right-click a sample class and select *Run As* > *2 JUnit Test*.

The sample runs, allowing you to find out the features of the BI Semantic Layer RESTful Web Service SDK.

5.2 About the Web Intelligence REST API Samples

The SDK samples described in the following table work with the following set of universes and documents:

- `Warehouse.unx`, `eFashion.unx`, and `eFashion.unv` universes
- `[Raylight Sample Template] Change Source`, `[Raylight Sample Template] Empty`, `[Raylight Sample Template] Refresh`, and `[Raylight Sample Template] Schedule` documents

You must deploy the LCMBIAR file included in the ZIP file in your CMS repository to be able to run the samples. The archive file is `lcmbiar/RaylightSamples.lcmbiar`. Once run, Web Intelligence documents are generated in the CMS repository and can be found under `Documents/Folders/Web Intelligence Samples`.

Class	Description
`CreateAndSaveSample`	Creates and saves an empty Web Intelligence document: • Creates a new empty document • Adds a data provider • Adds a query specification • Adds a report • Saves the document The data provider and query specification are attached to the document and available for future use. To display traces in the console, you can uncomment the line containing `deleteDocument(newDocID)`.
`ChartSample`	Adds four charts to an existing document: • Refreshes the document (data provider/query specification) • Adds a cell used to display the title • Adds a column chart to the report • Adds a donut chart to the report • Adds a bar chart with its own data filter • Adds a surface chart with its own data filter
`TableSample`	Adds a table with seven columns to a document: • Refreshes the document with prompt values • Adds a cell used to display the title • Adds a vertical table to the report • Resizes the product column width as automatic • Adds alerters and a variable • Adds an empty column to the table and fills it with the variable

Class	Description
`RefreshSample`	Refreshes a document through different ways: • Refreshes without changes • Refreshes with changes: custom context & prompt • Cancels a refresh
`ScheduleSample`	Schedules a document through different ways: • Purges the document from its data and saves it • Schedules the document to the BI launch pad inbox • Schedules the document to an email address • Schedules the document to an FTP server • Schedules the document to a file system Only the schedule to the BI launch pad does not require CMS server configuration.
`FreeHandSQLSample`	Uses the free-hand SQL data provider: • Creates an empty document • Renames the report • Checks if the connection is relational • Adds a free-hand SQL data provider using a custom query • Adds a cell title • Refreshes the data without any changes • Adds a chart using data from the free-hand SQL data provider • Adds an input control You must know the database schema before using a custom query.
`ChangeSourceSample`	Changes the data source of an existing document: • Gets the suggested mapping to change source • Applies the suggested mapping without any changes To edit the mapping and set up your own data objects, you can uncomment the line containing `editMapping()`.

resources and util Packages

The `com.sap.webi.raylight.samples.resources` package contains XML descriptions used in sample requests, such as the query specification. The `com.sap.webi.raylight.utils` package provides classes that help to perform utility actions in the samples (for example, login, fill in CMS parameters, send a request, and so on). The `com.sap.webi.raylight.param` contains the parameters required for sample execution.

To Deploy the LCMBIAR File [page 65]

To Deploy the Samples with Eclipse [page 65]

5.2.1 To Deploy the LCMBIAR File

You must deploy the LCMBIAR archive file before you deploy and run the samples.

1. Login to the CMC.
2. Select *Promotion Management* under the *Manage* menu.
3. Select *Import* > *Import file*.
4. Check *File System* and click *Choose File* to upload the `RaylightSamples.lcmbiar` file from the ZIP.
5. Click *OK*.
   The *New Job* dialog box appears. The new job name RSamples is prefilled.
6. Select *From File* for the *Source* and *Login to a New CMS* for *Destination*.
   The *Login To System* dialog box appears.
7. Type in the credentials of your CMS repository and click *OK*.
8. Click *Create* in the *New Job* dialog box.
   The LCMBIAR file content appears.
9. Click *Manage Dependencies* to check the dependencies between universes and documents. If OK, click *Apply & Close*.
10. Click *Promote* to add the job to the Promotion Job folder.
    The RSamples job appears in the Promotion Job folder.
11. Select this job and click *Promote*.
    All the objects to be promoted are listed.
12. Click *Schedule* to finish the deployment.

5.2.2 To Deploy the Samples with Eclipse

You can run Java samples in your Eclipse project.

1. Launch Eclipse by double-clicking the `eclipse.exe` file and select your workspace.

   Use Eclipse 3.6 or higher.
2. Create a project.
3. Select *File* > *Import*.
   The Import dialog box appears.
4. Expand *General*, select *Existing Projects into Workspace*, and click *Next*.
5. Select the *Select archive file* option and click *Browse*.
6. Browse to the sample directory and select `RaylightRESTWS_Samples.zip`.
7. In the *Projects* area, select *com.sap.webi.raylight.samples* and click *Finish*.
   The `com.sap.webi.raylight.samples` project folder appears in the *Package Explorer* view.
8. Under *com.sap.sl.webi.raylight.param*, open the `Config.java` file for editing.
9. Fill in the following parameters with the values corresponding to your installation:
   - `CMS_SERVER_URL`: the URL exposed by the server hosting the RESTful web services
   - `CMS_USER`: user name used in your samples to connect to the CMS repository
   - `CMS_PASSWORD`: password used in your samples to connect to the CMS repository

10. Under *com.sap.webi.raylight.samples*, right-click a sample class and select *Run As* > *2 JUnit Test*.

The sample runs, allowing you to find out the generated Web Intelligence documents in your CMS repository under `Documents/Folders/Web Intelligence Samples`.

6 Concepts and Schemas of the RESTful Web Service SDKs

The following sections present thorough descriptions of XML request and response schemas for the universes, queries, parameters, charts, schedules, and data source mappings.

The following BI Semantic Layer concepts are supported in this release:

- Dimensions, attributes, and measures
- Predefined filters
- Simple custom query filters and subquery filters
- Combined queries
- List of values
- Parameters

Report Structure [page 67]

Charts [page 69]

Date and Time Formats [page 104]

Queries [page 105]

Parameters [page 132]

Change Source [page 145]

Schedules [page 152]

Universes [page 157]

Object Full Paths [page 161]

User Rights [page 162]

Getting the Web Intelligence Security Rights for the Current User [page 163]

6.1 Report Structure

In a Web Intelligence document, the following items help identify the reports and report elements:

- Id
  The unique identifier of a report or report element inside a document. It is an integer.
- Reference
  The identifier of a report element with respect to the report it belongs to. It is a base64 string of the format "X.Y.A.[...]", where X is the report identifier, Y the report element identifier, and A the identifier of an instance of the report element. Instance identifiers are used to identify the same report element used several times in a report, for example a section in a report, or a cell in a table.

Where to Use References?

You use the reference of a report element to get:

- The node references of its corresponding map and the associated data. See Getting the Map of a Report [page 348].
- Its datapath. See Getting the Datapaths of a Report Element [page 412].
- Its dataset. See Getting the Dataset of a Report Element [page 416].

> **Note**
>
> You can use references or datapaths to retrieve the data associated with a specific instance of a report element.

Example

```
<report>
    <id>1</id>
    <name>Chart Demo</name>
    <reference>1.RS</reference>
    <showDataChanges>false</showDataChanges>
    ...
```

In the current document, the report identifier is "1" and the string "1.RS" is the reference of the report in the document.

```
<elements>
    <element type="Cell">
        <id>3</id>
        <reference>1.K.3</reference>
        ...
```

The report element identifier is "3". The string "1.K.3" represents the reference of the element in the report.

In addition, the report structure can be viewed as a tree, which means there is a parent/child relationship between reports and report elements. This mechanism allows to add an element in the report structure with accuracy.

Example

```
<elements>
    <element type="Cell">
        <id>14</id>
        <reference>1.E</reference>
        <name>Block 2</name>
        <parentId>1</parentId>
        ...
```

The report element of type "Cell" belongs to the report of which identifier is "1".

6.2 Charts

SAP BusinessObjects Web Intelligence offers end-users various possibilities to display data as graphical representations in their reports. These representations are called charts.

In the Web Intelligence semantics, charts can be viewed as report elements of which the type is `Visualization`. As for any report element, the REST Web Service SDK provides you with REST methods to:

- Create charts
- Retrieve chart details
- Update chart settings and expressions
- Delete charts

This section describes the XML grammar for any type of chart supported in this release.

Chart Types [page 69]

Chart Response Body Schema [page 70]

6.2.1 Chart Types

The following list describes the possible types of chart as returned by the call `GET <base_webi_REST_URL>/configuration/visualizations`. For a definition of each chart type, see the *SAP BusinessObjects Web Intelligence User's Guide*.

- Bar Series (HorizontalBar, HorizontalStackedBar, HorizontalStackedPercentBar, VerticalBar, VerticalStackedBar, VerticalStackedPercentBar, Bar3D)
- BoxPlot
- Dual and combined charts (DualBar, DualLine, CombinedBarLine, and DualCombinedBarLine)
- Funnel
- Gauges Series (AngularGauge, LinearGauge, Speedometer)
- Line
- Map Series (TreeMap, HeatMap)
- Pie Series (Pie, PieWithDepth, Doughnut)
- Point Series (Scatter, Bubble, PolarScatter, PolarBubble)
- Pyramid
- Radar
- Surface
- TagCloud
- Tile Series (Tile, TileDeviation)
- Waterfall

Related Information

Getting the Chart Types [page 271]

6.2.2 Chart Response Body Schema

Response Body Schema (XML)

(GET .../documents/<documentID>/reports/<reportID>/elements/<elementID>)

```
<element type="Visualization">
    ...
    <content>
        <chart type="string">
            <layout>
            <title>
            <legend>
            <dataLabels>
            <plotArea>
            <graphics>
            <axes>
```

The chart `type` attribute value is a string that specifies the type of chart [page 69].

Colors [page 70]

Layout [page 71]

Title [page 72]

Legend [page 73]

Data Labels [page 75]

Plot Area [page 78]

Graphics [page 86]

Axes [page 98]

6.2.2.1 Colors

You can apply colors to different areas of a chart such as titles, legends, labels, or axes. You define a color either with the RGB value or by using a gradient.

Fixed Colors

A color is expressed using the RGB color model and an opacity.

Attribute	Description	Value
`rgb`	The RGB color	The # sign followed by an hexadecimal value
`alpha`	The color opacity	[0, 255] from opaque to invisible

Example

```
<color rgb="#9d9d9d" alpha="187"/>
```

Gradient

A gradient of colors is defined by a starting color and an ending color. Each color has an RGB value and an opacity. The gradient is oriented according to the `orientation` attribute value (`Horizontal`|`HorizontalInverse`|`Vertical`|`VerticalInverse`|`DiagonalUp`|`DiagonalUpInverse`|`DiagonalDown`|`DiagonalDownInverse`).

Example

```
<gradient orientation="DiagonalDown">
    <start rgb="#ffffff" alpha="95"/>
    <end rgb="#000000" alpha="95"/>
</gradient>
```

6.2.2.2 Layout

The following table describes the `<layout>` attributes. All attributes are Boolean (`true`|`false`).

Attribute	Description
`showDimensionsWithEmptyMeasureValues`	Specifies whether the chart displays dimensions when some of their corresponding measures have no value
`showDimensionsWithMeasuresEqualToZero`	Specifies whether the chart displays dimensions when some of their corresponding measures have values equal to zero
`showDimensionsWithSumOfMeasuresEqualToZero`	Specifies whether the chart displays dimensions whose the corresponding sum of measure values is equal to zero
`showMeasuresWithEmptyDimensionValues`	Specifies whether the chart displays measures when some of their corresponding dimensions have no value

Attribute	Description
`showParentNodes`	Specifies whether the chart displays the parent of each dimension
`showMissingParent`	Specifies whether the chart displays a missing parent value as a node for the dimension
`showTotal`	Specifies whether the chart displays the sum of the measures
`parentAsTotal`	Specifies whether the chart displays the dimension parent for the sum of its measures
`duplicateRowAggregation`	Specifies whether the chart displays duplicate data. Measure values are not aggregated
`horizontal`	Specifies ther chart orientation (horizontal or vertical)

6.2.2.3 Title

```
<title visible="Boolean">
    <style>
    <label>
    <layout>
```

The `visible` Boolean attribute specifies whether the title should be displayed in the chart.

Element	Description
`<style>`	The style of a title defines the following properties: • `<border>`, which is the border thickness and color • `<background>`, which is the background color • `<font>`, which defines title font properties (size, face, italic, bold, strikethrough, underline, and RGB color) • `<alignment>`, which defines the text alignment (horizontal, vertical, and text policy). The text policy can be `Wrap`, `NoWrap` or `Truncate`.
`<label>`	The title itself. Can be text or formula.
`<layout>`	**Attributes**: • `location` (`Top|Bottom|Left|Right`) • `orientation` (`Auto|VerticalLettering`) • `spacing`, whose value range is [0, 8] • `adjust` (Boolean) If `adjust` is `true`, then horizontal and vertical proportionalities can be defined. Their possible types are: • `Auto`. No value has to be set. • `Fixed`, which defines a value in unit [page 391] • `Proportional`, which defines a ratio whose range is [0.0, 1.0] with a step of 0.01

Example

```
<title visible="true">
    <style>
        <border thickness="None">
            <color rgb="#000000" alpha="255"/>
        </border>
        <background>
            <color rgb="#000000" alpha="0"/>
        </background>
        <font size="9" face="Arial" italic="true" bold="true"
strikethrough="false" underline="true|false" rgb="#000000"/>
        <alignment horizontal="Left" vertical="Center" textPolicy="Truncate"/>
    </style>
    <label dataType="String">"My Title"</label>
    <layout location="Top" orientation="Auto" spacing="2" adjust="true">
        <horizontalProportionality type="Fixed">3.25</horizontalProportionality>
        <verticalProportionality type="Proportional">0.33</
verticalProportionality>
    </layout>
</title>
```

Related Information

Colors [page 70]

6.2.2.4 Legend

> **Note**
>
> The Waterfall chart has no legend.

```
<legend visible="Boolean" mode="string">
    <style>
    <title>
    <layout>
    <extraInfo>
```

Attribute	Type or Value	Description
`visible`	*Boolean*	Specifies whether the legend is displayed in the chart
`mode`	`Color|Size|ColorAndSize`	Only used in Bubble charts

Element	Description
`<style>`	The style of a legend has the following properties: • `<border>`, which defines the border thickness and color • `<background>`, which defines the background color • `<font>`, which defines legend font properties (size, face, italic, bold, strike-through, underline, and RGB color) • `<alignment>`, which defines the text alignment (horizontal, vertical, and text policy). The text policy can be `Wrap`, `NoWrap` or `Truncate`.
`<title>`	Defines the same properties as a chart title. See Title [page 72].
`<layout>`	**Attributes**: • `groupByDimension` (Boolean) • `symbolSize`, whose value range is [4, 32] • `location` (`Top|Bottom|Left|Right`) • `orientation` (`Auto|Horizontal|Vertical`) • `spacing`, whose value range is [0, 8] • `adjust` (Boolean) If `adjust` is `true`, then horizontal and vertical proportionalities can be defined. Their possible types are: • `Auto`. No value has to be set. • `Fixed`, which defines a value in unit [page 391] • `Proportional`, which defines a ratio whose range is [0.0, 1.0] with a step of 0.01
`<extraInfo>`	**Note** For BoxPlot, TreeMap, PieWithDepth, Bubble, PolarBubble, and TagCloud charts only. The `visible` Boolean attribute specifies whether additional information is displayed in the legend. The `<font>` child element defines font properties of additional information (size, face, italic, bold, strikethrough, underline, and RGB color).

Example

```
<legend visible="true">
    <style>
        <border thickness="None">
            <color rgb="#000000" alpha="0"/>
        </border>
        <background>
            <color rgb="#000000" alpha="0"/>
        </background>
        <font size="12" face="Helvetica" italic="true" bold="true"
strikethrough="false" underline="false" rgb="#00c7ff"/>
```

```
        <alignment horizontal="Left" vertical="Center" textPolicy="Wrap"/>
    </style>
    <title visible="true">
        <style>
            <border thickness="None">
                <color rgb="#000000" alpha="255"/>
            </border>
            <background>
                <color rgb="#000000" alpha="0"/>
            </background>
            <font size="6" face="Tahoma" italic="false" bold="true"
strikethrough="false" underline="true" rgb="#00ff00"/>
            <alignment horizontal="Left" vertical="Center"
textPolicy="Truncate"/>
        </style>
        <label dataType="String">="Legend Title"</label>
        <layout orientation="Auto" spacing="2"/>
    </title>
    <layout groupByDimension="false" symbolSize="7" location="Right"
orientation="Auto" spacing="4" adjust="true">
        <horizontalProportionality type="Fixed">3.25</horizontalProportionality>
        <verticalProportionality type="Proportional">0.33</
verticalProportionality>
    </layout>
    <extraInfo visible="true">
        <font size="8" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#707070"/>
    </extraInfo>
</legend>
```

Related Information

Colors [page 70]

6.2.2.5 Data Labels

> **Note**
>
> The BoxPlot and TagCloud charts have no data label.

```
<dataLabels type="string" visible="Boolean">
    <style>
    <layout>
    <format>
```

Attribute	Type or Value	
`type`	*string* (see values below)	The type of data caption to display
`visible`	*Boolean*	Specifies whether the labels should be displayed in the chart

The following table describes the possible `type` attribute values.

Attribute Value	Description
`Auto`	A label is added automatically.
`Value`	The data caption displays the value. This corresponds to a value axis.
`Label`	The data caption displays the category. This corresponds to a category axis.
`Percent`	The data caption displays the values in percent.
`LabelAndValue`	The data caption displays the category followed by the value.
`LabelAndPercent`	The data caption displays the category followed by the value in percent.
`XValue`	The data caption displays the second value corresponding to the second value axis.
`Weight`	The data caption displays the category followed by the value.
`Scale`	The data caption displays the value corresponding to the bubble width.

The charts do not display all data label types. The following table shows the possible types of data labels per type of chart.

Attribute Value	Type of chart
`Auto`	All except Waterfall
`Value`	All
`Label`	All except Line, DualLine, Surface, Radar, and HeatMap
`Percent`	All except Point series, Radar, HeatMap, and Waterfall
`LabelAndValue`	Pie series only
`LabelAndPercent`	Pie series only
`XValue`	Point series only
`Weight`	TreeMap only
`Scale`	Bubble and PolarBubble charts only

The following table shows the `<dataLabels>` children elements.

Element	Description
`<style>`	The style of a label defines the following properties: • `<border>`, which is the border thickness and color • `<background>`, which is the background color • `<font>`, which defines the label font properties (size, face, italic, bold, strikethrough, underline, and RGB color) • `<alignment>`, which defines the text alignment (horizontal, vertical, and text policy). The text policy can be `Wrap`, `NoWrap` or `Truncate`.

Element	Description
`<layout>`	**Attributes**: • `autoHiding`, which defines whether labels are hidden automatically when there is not enough space • `resolveOverlapping`, which defines whether the overlapped labels are resolved • `position` (`InsideFirstOutsideOtherwise|OutsideFirstInsideOtherwise|Inside|Outside`) • `orientation` (`Horizontal|Vertical`) • `spacing`, which defines the margin and whose value range is [0, 8] • `mode`, which defines the display mode for Pie charts only (`Side|Circular`) • `singleLine`, which defines whether the data label is on one single line (for `LabelAndValue` or `LabelAndPercent` type only) • `percentMinValue`, which defines the minimum percent value to display for Pie charts only. Value range is [0.0, 100.0] with a step of 0.01. Layout also shows the `<separator>` child element to define the starting and ending symbols to use in data labels in the case of a Pie chart. The string maximum length for a value is 256.
`<format>`	The expression to use to format and display tick values. **Note** For `Percent` or `LabelAndPercent` type only.

Example

```
<dataLabels type="LabelAndPercent" visible="false">
    <style>
        <border thickness="None">
            <color rgb="#000000" alpha="255"/>
        </border>
        <background>
            <color rgb="#000000" alpha="0"/>
        </background>
        <font size="8" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#707070"/>
        <alignment textPolicy="NoWrap"/>
    </style>
    <layout autoHiding="true" resolveOverlapping="false" mode="Side"
position="Outside" spacing="0" singleLine="true"
        percentMinValue="10.0">
        <separator>
            <start>(</start>
            <end>)</end>
        </separator>
    <format default="true" sample="1234,57" type="Custom">
        <template positive="0.00"/>
    </format>
</dataLabels>
```

Related Information

Colors [page 70]

6.2.2.6 Plot Area

> **Note**
>
> The HeatMap chart has no plot area.

```
<plotArea>
    <xSeries>
    <title>
    <background>
    <grids>
```

Element	Description
`<xSeries>`	Defines specific layout properties for each chart type except Pie charts.
`<title>`	Defines the same properties of a chart title. See Title [page 72]. **Note** The Bar3D, Map Series, PolarBubble, PolarScatter, BoxPlot, TagCloud, and Waterfall charts have no title in `<plotArea>`.

`<background>`	The background area can have one color (plain) or two colors (striped).
	• Plain
	`<background mode="Plain">`
	`  <coloring>`
	`  <color rgb="#000000" alpha="0"/>`
	`  </coloring>`
	`</background>`
	• Stripped
	`<background mode="Striped">`
	`  <coloring lightingAdjustment="0.9">`
	`  <color rgb="#00ff00" alpha="255"/>`
	`  </coloring>`
	`  <coloring lightingAdjustment="0.9">`
	`  <color rgb="#ff00ff" alpha="255"/>`
	`  </coloring>`
	`</background>`
	`lightingAdjustment` defines the lighting color adjustment value. Value range is [0.0, 1.0] with a step of 0.01.

Element	Description

`<grids>`	Defines the grid colors. Bidimensional charts have two grids (`Vertical` and `Horizontal`), while the 3DBar chart has three grids (`Vertical`, `Horizontal`, and `Depth`).
	
	`<grids>`
	`  <grid type="Vertical">`
	`  <color rgb="#000000" alpha="0"/>`
	`  </grid>`
	`  <grid type="Horizontal">`
	`  <color rgb="#e7e7e7" alpha="255"/>`
	`  </grid>`
	`  <grid type="Depth">`
	`  <color rgb="#000000" alpha="89"/>`
	`  </grid>`
	`</grids>`

Bar, Line, and Surface Series Layout [page 79]

Box Series Layout [page 80]

Map Series Layout [page 81]

Point Series Layout [page 82]

Radar Series Layout [page 83]

Tag Cloud Series Layout [page 84]

Waterfall Series Layout [page 85]

6.2.2.6.1 Bar, Line, and Surface Series Layout

The following table describes the attributes common to the all charts of the Bar, Line, and Surface series (`<barSeries>`, `<lineSeries>`, and `<surfaceSeries>` elements of the chart definition).

Attribute	Type or Value	Description
`dashedLines`	*Boolean*	Specifies whether lines are displayed as dashed lines
`invertSuperimpositionOrder`	*Boolean*	Specifies whether the superimposition order of the layers plotted to each data series is inverted. In the case where lines or surfaces are stacked, this setting has no effect.
`spacingBetweenGroups`	*double*	The space between groups. Value range is [-1.0, 1.0] with a step of 0.01.
`spacingWithinGroups`	*double*	The space within groups. Value range is [-1.0, 1.0] with a step of 0.01.

The following attributes are specific to the Bar3D chart.

Attribute	Type or Value	Description
`showFloor`	*Boolean*	Specifies whether a virtual 3D floor is displayed
`showFirstEdge`	*Boolean*	Specifies whether the 3D wall first edge is displayed
`showSecondEdge`	*Boolean*	Specifies whether the 3D wall second edge is displayed

Example: Bar Series

```
<plotArea>
    <barSeries dashedLines="false" spacingBetweenGroups="0.2"
spacingWithinGroups="0.2"/>
    <title visible="true">
        <style>
            <border thickness="None">
                <color alpha="181" rgb="#ff0000"/>
            </border>
            <background>
                <color alpha="115" rgb="#000000"/>
            </background>
            <font size="10" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#ff00ff"/>
        </style>
        <layout spacing="0"/>
    </title>
    <background mode="Plain">
        <coloring>
            <color alpha="0" rgb="#000000"/>
        </coloring>
    </background>
    <grids>
        <grid type="Vertical">
            <color alpha="0" rgb="#000000"/>
        </grid>
        <grid type="Horizontal">
            <color alpha="255" rgb="#e7e7e7"/>
        </grid>
    </grids>
</plotArea>
```

6.2.2.6.2 Box Series Layout

The following table describes the attributes specific to the BoxPlot chart definition (`<boxSeries>`).

Attribute	Type or Value	Description
`dashedLines`	*Boolean*	Specifies whether lines are displayed as dashed lines

Attribute	Type or Value	Description
`hideOutliers`	*Boolean*	Specifies whether outlier points are hidden
`spacingBetweenGroups`	*double*	The space between groups. Value range is [-1.0, 1.0] with a step of 0.01.

Example

```
<plotArea>
    <boxSeries dashedLines="true" hideOutliers="true"
spacingBetweenItems="0.2"/>
    <background mode="Plain">
        <coloring>
            <color alpha="0" rgb="#000000"/>
        </coloring>
    </background>
    <grids>
        <grid type="Vertical">
            <color alpha="0" rgb="#000000"/>
        </grid>
        <grid type="Horizontal">
            <color alpha="255" rgb="#e7e7e7"/>
        </grid>
    </grids>
</plotArea>
```

6.2.2.6.3 Map Series Layout

```
<mapSeries>
    <zoneTitle>
        <font>
        <alignment>
        <layout>
```

The following table describes the attributes specific to the TreeMap chart definition.

Attribute	Type or Value	Description
`showTreeMapRoot`	*Boolean*	Specifies whether the root node of the hierarchy is displayed
`depth`	*integer*	The depth for 3D look in proportion to chart size. Value range is [-1, 16].
`fixParentWeight`	`Fix|Strict`	Enables or disables data auto-correction in hierarchical data
`hierarchicalView`	*Boolean*	Specifies whether zones are displayed in a hierarchical view

The `<zoneTitle>` element has the `visible` Boolean attribute that defines whether the title of the zone should be displayed in the chart.

The following table shows the `<zoneTitle>` children elements.

Element	Description
`<font>`	The font properties of the zone title (size, face, italic, bold, strikethrough, underline, and RGB color)
`<alignment>`	The text alignment (horizontal, vertical, and text policy). The text policy can be `Wrap`, `NoWrap` or `Truncate`.
`<layout>`	**Attributes**: • `orientation` (`Auto|VerticalLettering`) • `spacing`, whose value is in unit [page 391] and the metric range is [0, 1500]

Example

```
<plotArea>
    <mapSeries showTreeMapRoot="true" depth="-1" fixParentWeight="Fix"
hierarchicalView="true">
        <zoneTitle visible="true">
            <font size="8" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#ffffff"/>
            <alignment horizontal="Center" vertical="Center"
textPolicy="Truncate"/>
            <layout orientation="Auto" spacing="0.417"/>
        </zoneTitle>
    </mapSeries>
</plotArea>
```

6.2.2.6.4 Point Series Layout

The following table describes the attributes common to all charts of the Point series (`<pointSeries>`).

Attribute	Type or Value	Description
`dashedLines`	*Boolean*	Specifies whether lines are displayed as dashed lines
`invertSuperimpositionOrder`	*Boolean*	Specifies whether the superimposition order of the layers plotted to each data series is inverted. In the case where lines or surfaces are stacked, this setting has no effect.

The following attributes are specific to the Bubble and PolarBubble charts.

Attribute	Type or Value	Description
`scale`	*integer*	The ratio between the plot area size and the biggest bubble diameter. Value range is [2, 10].
`scalingMode`	`Proportional|Perceptual`	The scaling mode

Example

```
<plotArea>
    <pointSeries dashedLines="false" invertSuperimpositionOrder="false"
scale="5" scalingMode="Perceptual"/>
    <background mode="Striped">
        <coloring lightingAdjustment="0.9">
            <color alpha="110" rgb="#ffffce"/>
        </coloring>
        <coloring lightingAdjustment="0.9">
            <color alpha="115" rgb="#ceffff"/>
        </coloring>
    </background>
    <grids>
        <grid type="Vertical">
            <color alpha="0" rgb="#000000"/>
        </grid>
        <grid type="Horizontal">
            <color alpha="255" rgb="#e7e7e7"/>
        </grid>
    </grids>
</plotArea>
```

6.2.2.6.5 Radar Series Layout

The following table shows the attributes specific to the Radar chart definition (`<radarSeries>`).

Attribute	Type or Value	Description
`invertSuperimpositionOrder`	*Boolean*	Specifies whether the superimposition order of the layers plotted to each data series is inverted. In the case where lines or surfaces are stacked, this setting has no effect.
`bringToFront`	*Boolean*	Specifies whether the grid is in front of the data
`polygonal`	*Boolean*	Specifies whether the grid is displayed as polygonal lines as opposed to circles

Example

```
<plotArea>
    <radarSeries invertSuperimpositionOrder="true" bringToFront="true"
polygonal="true"/>
    <background mode="Plain">
        <coloring>
            <color alpha="89" rgb="#ffffce"/>
        </coloring>
    </background>
    <grids>
        <grid type="Vertical">
            <color alpha="87" rgb="#000000"/>
        </grid>
        <grid type="Horizontal">
            <color alpha="87" rgb="#007d00"/>
        </grid>
    </grids>
</plotArea>
```

6.2.2.6.6 Tag Cloud Series Layout

```
<tagCloudSeries>
    <tag>
        <font>
```

The following tables describe the attributes specific to the TagCloud chart definition (`<tagCloudSeries>`).

Attribute	Type or Value	Description
`levelColoring`	*integer*	The depth of coloring areas. Value range is [0, 64].
`comparator`	`Weight|Names|Rating`	The dataset characteristic used to compare words and choose their location
`mode`	`Row|Column|Wordle`	The tag display mode
`alignment`	`Left|Center|Right|Justify`	The text alignment
`orientation`	`MainlyHorizontal|HorizontalAndVertical|MainlyVertical|HorizontalOnly|VerticalOnly`	The text orientation mode
`fillRate`	*double*	The screen fill rate with words. Value range is [0.100, 1.500] with a step of 0.025.

Attribute	Type or Value	Description
`spacingBetweenTags`	`Auto|Fixed`	The space between tags
`spacingValue`	*double*	The space value in unit [page 391] if `spacingBetweenTags` is `Fixed`.

The following table describes the `<tag>` attributes.

Attribute	Type or Value	Description
`maxFontRatio`	`Auto|Fixed`	The method to choose maximum font size of words
`maxFontSize`	*integer*	The word maximum font size if `maxFontRatio` is `Fixed`. Value range is [1, 56].
`minFontRatio`	`Auto|Fixed`	The method to choose minimum font size of words
`minFontSize`	*integer*	The word minimum font size if `minFontRatio` is `Fixed`. Value range is [1, 256].
`minVisibleFontSize`	*integer*	The font size threshold under which words are removed. Value range is [1, 50].
`fontScaling`	`Linear|Exponential|Logarithmic`	The word font size curve compared to word order

`<font>` defines the title font properties (face, italic, bold, strikethrough, underline, and RGB color). Font size is ignored. See Title [page 72].

Example

```
<plotArea>
    <tagCloudSeries levelColoring="45" comparator="Weight" mode="Wordle"
orientation="HorizontalAndVertical"
        fillRate="0.65" spacingBetweenTags="Fixed" spacingValue="0.097">
        <tag maxFontRatio="Auto" minFontRatio="Auto" minVisibleFontSize="4"
fontScaling="Logarithmic">
            <font size="6" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#555555"/>
        </tag>
    </tagCloudSeries>
</plotArea>
```

6.2.2.6.7 Waterfall Series Layout

```
<waterfallSeries>
    <line>
        <color>
```

The following table describes the attributes specific to the Waterfall chart definition.

Attribute	Type or Value	Description
`dashedLines`	*Boolean*	Specifies whether lines are displayed as dashed lines
`referenceLine`	*Boolean*	The reference line
`spacingBetweenItems`	*double*	The space between groups. Value range is [-1.000 , 1.000], with a step of 0.01.

`<line>` thickness is defined via the `width` attribute with a range of [1, 7] and the color by using a `<color>` child element.

Example

```
<plotArea>
    <waterfallSeries dashedLines="false" referenceLine="true"
spacingBetweenItems="0.46">
        <line width="7">
            <color alpha="255" rgb="#ff00ff"/>
        </line>
    </waterfallSeries>
    <background mode="Plain">
        <coloring>
            <color alpha="0" rgb="#000000"/>
        </coloring>
    </background>
    <grids>
        <grid type="Vertical">
            <color alpha="0" rgb="#000000"/>
        </grid>
        <grid type="Horizontal">
            <color alpha="255" rgb="#e7e7e7"/>
        </grid>
    </grids>
</plotArea>
```

6.2.2.7 Graphics

```
<graphics>
    <coloring>
    <rendering>
    <effects>
```

Element	Description

`<coloring>`	Specifies the palette and coloring method used by the chart. Both built-in [page 276] or custom [page 279] palettes can be used.
	
	Most of the charts can define only one palette. The `refId` attribute value is a valid palette identifier.
	
	`<coloring>`
	`  <palettes>`
	`  <palette alpha="0" refId="green"/>`
	`  </palettes>`
	`</coloring>`
	
	Dual charts can define two palettes:
	
	`<coloring>`
	`  <palettes method="Dual">`
	`  <palette type="Primary" refId="blue"/>`
	`  <palette type="Secondary" refId="5f95e34e-b5c1-49d5-ac60-eb73ee5527c2"/>`
	`  </palettes>`
	`</coloring>`
	
	Point, Map, TagCloud and Waterfall Series have specific coloring methods. See Colorings [page 88].

`<rendering>`	**Attributes** for common rendering:
	
	• `filter`, which defines the light and shadow rendering filter (`None`|`SimpleLighting`|`SimpleShadows`|`SimpleLightingAndShadows`|`RealLighting`|`ComplexShadows`|`RealLightingAndComplexShadows`|`ImageEmbossed`)
	• `look3D` (Boolean)
	
	`<rendering filter="SimpleLightingAndShadows" look3D="true">`
	
	Attributes specific to Bar3D charts:
	
	• `shading` is the 3D lighting mode (`Faceted`|`Smooth`)
	• `shape` is the 3D bar shape (`Box`|`RoundedBox`|`Cylinder`)
	
	`<rendering filter="ImageEmbossed" shading="Smooth" shape="RoundedBox">`
	
	You can set further options depending on the chart type. See Rendering [page 91].

`<effects>`	The rendering effects of the graphics. They are specific to each chart series. See Effects [page 93]. ⓘ Note TagCloud charts have no additional effects.

6.2.2.7.1 Colorings

Point Series

The following table shows the `<pointSeries>` children elements.

Element	Description
`<colorGroup>`	The color of a group **Attributes**: • `type` (`Fixed|Deepest`) • `depth`, whose minimum is `1`
`<shapeGroup>`	The shape of a group **Attributes**: • `type` (`Fixed|Deepest`) • `depth`, whose minimum is `1`

Example

```
<coloring>
    <palettes>
        <palette alpha="0" refId="SAP Standard 2011"</>
    </palettes>
    <pointSeries>
        <colorGroup type="Fixed" depth="1"/>
        <shapeGroup type="Deepest"/>
    </pointSeries>
</coloring>
```

Map Series and TagCloud Series

Some specific coloring methods are provided for the Map Series and the TagCloud chart (`<mapSeries>` and `<tagCloudSeries>`).

Attribute	Type or Value	Description
`method`	`Palette|GradientBased|GradientBasedNeutralPolarity|CustomRange`	Defines the coloring method
`useInternalPalette`	*Boolean*	Specifies whether the built-in palette is used

The following table describes their common children elements.

Element	Description
`<gradientPalette>`	Defines the gradient color palette if `method="GradientBased"`. **Attributes**: • `type`, which defines the number of colors in gradient. Possible values are `Colors2` for two colors and `Colors3` for 3 colors. • `start`, which defines the first color of the gradient color palette • `middle`, which defines the middle color of the gradient color palette • `end`, which defines the last color of the gradient color palette
`<outOfRange>`	The color of the out of ranges values
`<nullOrEmpty>`	The color of the null or empty values
`<measurePolarityGradientPalette>`	Defines the gradient palette if `method="GradientBasedNeutralPolarity"`. **Attributes**: • `type` • `start` • `middle` • `end`
`<data>`	**Attributes**: • `distributionMode` that defines how data is distributed in the coloring zone (`ByValues`|`ByQuantiles`) • `intervalSyntax` that defines how data intervals display in the legend (`Basic`|`US`|`ISO31-11`)
`<ranges>`	Defines the zones and the associated colors. **Attributes** if `method` is **not** `"CustomRange"`: • `number` ([0, 64]) • `from` that defines the beginning of the interval values from which the color method is applied • `to` that defines the end of the interval values to which the color method is applied **Attributes** if `method="CustomRange"`: `percentage`(Boolean) **Child element** if `method="CustomRange"`: `<range>`.
`<range>`	Defines the minimum and maximum values for the color of the zone defined by the interval if `method="CustomRange"`. **Attributes**: • `from` that defines the beginning of the interval values from which the custom color method is applied • `to` that defines the end of the interval values to which the custom color method is applied

Example: Map Series

```
<coloring>
    ...
    <mapSeries method="CustomRange" useInternalPalette="true">
        <outOfRange>
            <color rgb="#c0c0c0" alpha="117"/>
        </outOfRange>
        <nullOrEmpty>
            <color rgb="#e0e0e0" alpha="125"/>
        </nullOrEmpty>
        <data distributionMode="ByQuantiles" intervalSyntax="ISO31-11"/>
        <ranges percentage="true">
            <range from="0.0" to="33.0">
                <color rgb="#ff0000" alpha="255"/>
            </range>
            <range from="33.0" to="67.0">
                <color rgb="#000000" alpha="255"/>
            </range>
            <range from="67.0" to="100.0">
                <color rgb="#00ff00" alpha="255"/>
            </range>
        </ranges>
    </mapSeries>
</coloring>
```

Example: Tag Cloud Series

```
<coloring>
    ...
    <tagCloudSeries method="Palette">
        <outOfRange>
            <color rgb="#c0c0c0" alpha="117"/>
        </outOfRange>
        <nullOrEmpty>
            <color rgb="#e0e0e0" alpha="125"/>
        </nullOrEmpty>
        <data distributionMode="ByQuantiles" intervalSyntax="ISO31-11"/>
        <ranges number="5" from="-4.497" to="220000.0"/>
    </tagCloudSeries>
</coloring>
```

Waterfall Series

`<waterfallSeries>` contains all settings to configure the color of some specified values. Each of the following `<waterfallSeries>` children elements has the `type` attribute (`Auto|Fixed`). If `type` is `Fixed`, then a `<color>` has to be set. Gradients of colors are allowed for the start and total values only.

Element	Description
`<start>`	The color of the start value

Element	Description
`<total>`	The color of the total value
`<negative>`	The color of the negative values
`<positive>`	The color of the positive values

Example

```
<coloring>
    ...
    <waterfallSeries>
        <start type="Fixed">
            <color rgb="#00ff00" alpha="130"/>
        </start>
        <total type="Fixed">
            <color rgb="#0000ff" alpha="110"/>
        </total>
        <negative type="Auto"/>
        <positive type="Auto"/>
    </waterfallSeries>
</coloring>
```

6.2.2.7.2 Rendering

Edge

`<edge>` allows you to configure the borders of the chart elements. The following table describes the `<edge>` attributes.

Attribute	Type or Value	Description
`type`	`None|Default|Override|Auto`	The border type
`width`	*integer*	The line thickness that you can set to Bar3D charts. Value range is [1, 7].

You can also set a color to an edge by using the `<color>` child element.

Rotation

The following table shows the charts that support the rotation of the graph and the corresponding `<rotation>` attributes.

Chart	<rotation> Attributes
Bar3D	The rotation angle values for the 3D orientation: • `xAngle` on the X axis. Value range is [-90, 90]. • `yAngle` on the Y axis. Value range is [-180, 180].
Pie series	• `startAngle` defines the start angle of the Pie in degrees. Value range is [0, 360] with a step of 1. • `clockwise` is a Boolean that defines the Pie rotation
Polar Bubble and Polar Scatter	• `startAngle` defines the start angle of the chart in degrees. Possible values are `0`, `90`, `180`, `270`, and `360`. • `clockwise` is a Boolean that defines the chart rotation
Radar	`clockwise` is a Boolean that defines the Radar rotation

Shadow

`<shadow>` allows you to configure the shadow properties of a chart element. The following table describes the `<shadow>` attributes.

Attribute	Type or Value	Description
`effect`	`None|OneSided`	Drops a shadow behind the chart elements. Possible values
`xOffset`	*double*	The horizontal distance or offset of the shadow from each chart element in unit [page 391].
`yOffset`	*double*	The vertical distance, or offset of the shadow from each chart element in unit [page 391].
`filterPassCount`	*integer*	The complexity of the effect. When a visual complexity is increased, performance may be decreased. Value range is [1, 9] with a step of 1.
`filterWindowSize`	*integer*	The smoothness of the shadow. The higher the value, the smoother the effect. Value range is [3, 9] with a step of 2.
`lightPower`	*double*	The light intensity.Value range is [-1.000, +1.000] with a step of 0.05.

You can also set a color to a shadow by using the `<color>` child element.

Example

```
<rendering>
    <edge type="Override" width="7">
        <color rgb="#ff00ff" alpha="189"/>
    </edge>
    <rotation startAngle="15" clockwise="true"/>
    <shadow effect="OneSided" xOffset="0.0" yOffset="0.0" filterPassCount="3"
filterWindowSize="5" lightPower="0.3">
        <color rgb="#9d9d9d" alpha="187"/>
    </shadow>
</rendering>
```

6.2.2.7.3 Effects

`<effects>` allows you to configure special effects for some chart series.

- Pie Series [page 93]
- Bar Series, Box Chart, and Waterfall Series [page 94]
- Line Series, Radar Series [page 95]
- Surface Series [page 96]
- Point Series [page 97]
- Map Series [page 97]
- Combined Charts [page 97]

Pie Series

`<pieSeries>` allows you to configure special effects for the Pie charts. The following table describes the `<pieSeries>` attributes.

Attribute	Type or Value	Description
`effect`	`None|Halo|Glossy|Button3D|ImageEmbossed|Moonlight`	The type of effect applied to the Pie chart
`material`	`None|BrushedMetal1|BrushedMetal2|Water|Arabesque`	The material applied to the chart texture
`depth`	*double*	The Pie chart depth expressing a third value in unit [page 391].
`outerRadius`	*integer*	The radius of the doughnut hole that you can set to Pie Doughnut charts only. Value range is [1, 99] .

Example

```
<effects>
    <pieSeries effect="Button3D" material="Water" depth="1.208"/>
</effects>
```

Bar Series, Box Chart, and Waterfall Series

The following table describes the attributes common to `<barSeries>`, `<boxSeries>`, and `<waterfallSeries>`.

Attribute	Type or Value	Description
`effect`	`None|Volume|Gradient|Glossy|Cylinder|LightGlossy`	The type of effect applied to the chart
`progressiveAlpha`	*Boolean*	Specifies whether there is progressive transparency along bars
`roundedCorners`	*Boolean*	Specifies whether bar corners are rounded
`effectWidth`	*double*	The width ratio value for the volume effect. Value range is [0.000, 0.500] with a step of 0.01.
`colorBrighter`	*double*	The lighting color adjustment value to apply when displaying volume effects on top of lines. Value range is [1.000, 2.000] with a step of 0.01.
`colorLessBright`	*double*	The lighting color adjustment value applied when displaying a volume effect on bottom of lines. Value range is [0.000, 1.000] with a step of 0.01.

Example: Bar Series

```
<effects>
    <barSeries effect="None" roundedCorners="false"/>
</effects>
```

Example: Box Series

```
<effects>
    <boxSeries effect="Cylinder"/>
```

```
</effects>
```

Example: Waterfall Series

```
<effects>
    <waterfallSeries effect="Gradient" progressiveAlpha="true"/>
</effects>
```

Line Series, Radar Series

```
<effects>
    <lineSeries>
        <marker>
            <color>
            <edge>
                <color>
            <symbols>
                <symbol>
```

The following table describes the attributes common to `<lineSeries>` and `<radarSeries>`. However, the `style` attribute is specific to `<lineSeries>`.

Attribute	Type or Value	Description
`effect`	`None|Volume`	The type of effect applied to the chart
`width`	*integer*	The line thickness. Value range is [0, 7].
`style`	`Solid|Dot|Dash|DashDot`	Specifies whether the line is plain, dotted or dashed. Not used in Radar series.
`spline`	*Boolean*	Specifies whether data points are connected using cardinal spline curves (as opposed to straight lines)
`colorBrighter`	*double*	The lighting color adjustment value to apply when displaying volume effects on top of lines. Value range is [1.000, 2.000] with a step of 0.01.
`colorLessBright`	*double*	The lighting color adjustment value applied when displaying a volume effect on bottom of lines. Value range is [0.000, 1.000] with a step of 0.01.
`transparency`	*integer*	The transparency factor of the area defined by the lines in the Radar charts only. Value range is [0, 55].

`<marker>` defines the symbol to display at the measure points of the chart. The following table describes the `<marker>` attributes.

Attribute	Type or Value	Description
`visible`	*Boolean*	Specifies whether the marker is displayed
`size`	*integer*	The marker size. Value range is [4, 32].
`autoResize`	*Boolean*	Specifies whether the marker symbols are resized automatically according to the chart size

The following table describes the `<marker>` children elements.

Element	Description
`<color>`	The symbol color
`<edge>`	The marker border. A color or a gradient of colors can also be defined. **Attribute**: `type` (`None`|`Default`|`Override`|`Auto`) **Child Element**: `<color>`
`<symbols>`	The symbols set used as markers **Child Element**: `<symbol>` (`Circle`|`Star`|`Diamond`|`Square`)

Example

```
<effects>
    <lineSeries effect="Volume" style="Dash" width="7" spline="true"
colorBrighter="1.5" colorLessBright="0.7">
        <marker visible="true" size="32">
            <edge type="Override">
                <color alpha="255" rgb="#ff00ff"/>
            </edge>
            <symbols>
                <symbol>Circle</symbol>
                <symbol>Star</symbol>
                <symbol>Diamond</symbol>
                <symbol>Square</symbol>
            </symbols>
        </marker>
    </lineSeries>
</effects>
```

Surface Series

The only `<surfaceSeries>` attribute is `progressiveAlpha` that specifies whether there is progressive transparency along the surface.

Point Series

The following table describes the `<pointSeries>` attributes.

Attribute	Type or Value	Description
`effect`	`None|Halo|Glossy|Button3D|ImageEmbossed|Moonlight`	The type of effect applied to the chart
`showCenter`	*Boolean*	Specifies whether a cross marker is displayed at the center of the symbol

A `<marker>` is also defined to configure the symbol used at the measure points of the chart. See the section above for details.

Example

```
<effects>
    <pointSeries effect="Moonlight" showCenter="false">
        <marker visible="true" size="16">
            <color alpha="38" rgb="#ff00ff"/>
            <edge type="Override">
                <color alpha="181" rgb="#000000"/>
            </edge>
            <symbols>
                <symbol>Star</symbol>
                <symbol>Square</symbol>
                <symbol>Circle</symbol>
                <symbol>Diamond</symbol>
            </symbols>
        </marker>
    </pointSeries>
</effects>
```

Map Series

The only `<mapSeries>` attribute is `zoneEmboss` that specifies whether an embossed zone is displayed.

Combined Charts

Combined charts can have the effects of more than one series. For example, the DualCombinedBarLine and CombinedBarLine charts combine the effects of both Bar and Line Series.

Example: DualCombinedBarLine

```
<effects>
    <barSeries effect="None" roundedCorners="false"/>
    <lineSeries effect="None" width="2" spline="true">
        <marker visible="true" size="8">
            <edge type="Override">
                <color alpha="125" rgb="#00ff00"/>
            </edge>
            <symbols>
                <symbol>Circle</symbol>
                <symbol>Star</symbol>
                <symbol>Diamond</symbol>
                <symbol>Square</symbol>
            </symbols>
        </marker>
    </lineSeries>
</effects>
```

6.2.2.8 Axes

```
<axes>
    <axis role="string" visible="Boolean" optional="Boolean">
        <id>
        <name>
        <title>
        <layout>
        <coloring>
        <grid>
        <tick>
        <labels>
        <stacking>
        <scaling>
        <expressions>
    ...
```

Attribute	Type or Value	Description
`role`	*string*	The unique role of the axis. See Roles and Identifiers [page 101] for values.
`visible`	*Boolean*	Specifies whether the axis is displayed in the chart
`optional`	*Boolean*	Specifies whether you must assign expressions to the axis

Element	Description
`<id>`	The axis identifier. This positive integer is specific to each chart axis. **Note** You can use it to assign expressions to or unassign them from the axis. See Updating the Expressions of an Axis [page 446].
`<name>`	The internal name of the axis as a string. Cannot be changed.

Element	Description
`<title>`	The axis Title [page 72]. In addition, the `<separator>` element defines the character used as label separator in the axis title when no custom label is defined. Maximum length is 256.
`<layout>`	**Attributes**: • `reverseOrder`, a Boolean that defines whether labels order is reversed on the category axis • `continuous`, a Boolean that defines the layout of the category axis in a continuous mode • `adjust` (Boolean) If `adjust` is `true`, then horizontal and vertical proportionalities can be defined. Their possible types are: • `Auto`. No value has to be set. • `Fixed`, which defines a value in unit [page 391] • `Proportional`, which defines a ratio whose range is [0.0, 1.0] with a step of 0.01
`<coloring>`	The axis Colors [page 70].
`<grid>`	The color and background color of the grid around axis labels. See Colors [page 70]. ⓘ Note For category axes only.
`<tick>`	**Attributes**: • `color` • `length`, whose value range is [2, 8] • `margin` between the tick and its label, with range [0, 8] • `densityMode`, that can be either `Auto` or `Fixed` • `density`, whose value range is [0, 4] if `densityMode` is `Fixed`
`<labels>`	The axis Labels [page 102].
`<stacking>`	**Attributes** of the axis stacking mode: • `mode` (`Unstacked`|`Stacked`|`GloballyStacked`) • `stacked100percent` (`true`|`false`) ⓘ Note For value axes only.
`<scaling>`	The axis scaling mode used to determine the axis ticks. See Scaling [page 103]. ⓘ Note For value axes only.

Element	Description
`<expressions>`	The formulas assigned to the axis. The number of expressions depends on the chart type and axis role. For example, you can assign only one expression per axis in the TagCloud chart. **Attribute** of an expression of a Bar, Line, Surface, Dual, or Combined chart: • `regionType` (`Default`|`Bars`|`Lines`|`Surfaces`) **Attribute** of an expression of a HeatMap, TreeMap, or TagCloud chart for the measures on value axes: • `polarity` (`Auto`|`Ascending`|`Descending`|`Neutral`)

Example: Value Role Axis

```
<axis role="Value" visible="true" optional="false">
    <id>2</id>
    <name>Value Axis 1</name>
    <title visible="true">
        <style>
            <border thickness="None">
                <color rgb="#000000" alpha="255"/>
            </border>
            <background>
                <color rgb="#000000" alpha="0"/>
            </background>
            <font size="8" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#000000"/>
        </style>
        <layout spacing="0"/>
        <separator>&amp;amp;</separator>
    </title>
    <layout adjust="false"/>
    <coloring>
        <color rgb="#707070" alpha="255"/>
    </coloring>
    <tick length="5" margin="4" densityMode="Fixed" density="2"/>
    <labels visible="true" orientation="Auto" staggered="false">
        <font size="8" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#707070"/>
        <numberFormat default="true" sample="1,234567E3" type="Custom">
            <template positive="SCIENTIFIC"/>
        </numberFormat>
    </labels>
    <stacking mode="GloballyStacked" stacked100percent="false"/>
    <scaling mode="Linear" unitScaleFactor="0" roundMinMaxValues="false">
        <minValue type="Auto"/>
        <maxValue type="Auto"/>
    </scaling>
    <expressions>
         <formula regionType="Default" dataType="Numeric"
dataObjectId="DP0.DO7">=[Revenue]</formula>
    </expressions>
</axis>
```

Example: Category Role Axis

```
<axis role="MainCategory" visible="true" optional="false">
    <id>0</id>
    <name>Main Category Axis</name>
    <title visible="true">
        <style>
            <border thickness="None">
                <color alpha="255" rgb="#000000"/>
            </border>
            <background>
                <color alpha="0" rgb="#000000"/>
            </background>
            <font size="8" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#000000"/>
        </style>
        <layout spacing="0"/>
        <separator>&amp;amp;</separator>
    </title>
    <layout reverseOrder="false" continuous="false" adjust="false"/>
    <coloring>
        <color alpha="255" rgb="#707070"/>
    </coloring>
    <grid>
        <color alpha="255" rgb="#dadada"/>
        <background>
            <color alpha="0" rgb="#000000"/>
        </background>
    </grid>
    <tick length="4" margin="0"/>
    <labels visible="true" orientation="Auto" deleteMode="Auto"
autoResize="false">
        <font size="8" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#707070"/>
    </labels>
    <expressions>
        <formula dataType="String" dataObjectId="DP0.DO39">=[Country]</formula>
        <formula dataType="String" dataObjectId="DP0.DO31">=[Year]</formula>
    </expressions>
</axis>
```

6.2.2.8.1 Roles and Identifiers

A chart has a certain number of axes and each axis has a certain role. The following table describes the axis role per identifier (from `0` to `5`) for every chart or chart series.

Chart	0	1	2	3	4	5
Bar Series, Line, Surface, and CombinedBarLine	`Color`	`Category`	`Value`			
Pie	`PieSectorSize`	`PieSectorColor`				
PieWidthDepth and Doughnut	`PieSectorSize`	`PieDepthSize`	`PieSectorColor`			

Chart	0	1	2	3	4	5
Scatter	`Value1`	`Value2`	`Color`	`Shape`		
Bubble	`Value1`	`Value2`	`BubbleWidth`	`BubbleHeight`	`Color`	`Shape`
PolarScatter	`AngularValue`	`RadialValue`	`Color`	`Shape`		
PolarBubble	`AngularValue`	`RadialValue`	`BubbleWidth`	`Color`	`Shape`	
BoxPlot	`Category`	`Value`	`Color`			
Radar	`Color`	`Category`	`Shape`	`Value`		
HeatMap	`MainCategory`	`Color`	`SecondCategory`			
TreeMap	`RectangleTitle`	`RectangleWeight`	`Color`			
TagCloud	`Category`	`TagsWeight`	`TagsFamily`			
Waterfall	`Category`	`Value`				
Dual	`Category`	`Value1`	`Value2`	`Color`		

> **Note**
>
> There can only be up to two dimensional axes in Radar charts. For example, if data is displayed on "`Color`" and "`Category`", then you cannot assign data to "`Shape`".

6.2.2.8.2 Labels

```
<labels>
    <font>
    <numberFormat>
        <template>
```

Attribute	Type or Value	Description
`visible`	*Boolean*	Specifies whether a label is displayed on the axis

Attribute	Type or Value	Description
`orientation`	`Auto|Angle30|Angle60|VerticalLettering`	The label orientation **Note** You can use `Angle30` and `Angle60` on value axes only and if the chart is not Bar3D. You can use `Auto` and `VerticalLettering` in all other cases.
`deleteMode`	`Auto|Fixed`	The label deletion mode
`deleteFactor`	*integer*	The deletion factor if `deleteMode` is `Fixed`. Minimum is `1`.
`autoResize`	*Boolean*	Specifies whether font size is reduced automatically to display all labels
`staggered`	*Boolean*	Specifies whether labels are staggered
`textPolicy`	`NoWrap|Wrap|Truncate`	The label character policy if `staggered` is `true`

Element	Description
`<font>`	The label font properties (size, face, italic, bold, strikethrough, underline, and RGB color)
`<numberFormat>`	The format of the tick values defined through a template and a series of properties. **Note** For value axes only.

Example

```
<labels visible="true" orientation="Auto" staggered="false">
    <font size="8" face="Arial" italic="false" bold="true" strikethrough="false"
underline="false" rgb="#707070"/>
    <numberFormat default="true" sample="1,234567E3" type="Custom">
        <template positive="SCIENTIFIC"/>
    </numberFormat>
</labels>
```

6.2.2.8.3 Scaling

```
<scaling>
```

```
    <minValue>
    <maxValue>
```

Attribute	Type or Value	Description
`mode`	`Linear|Exponential|Logarithmic`	The scaling mode
`originInRange`	`Always|Auto|Never`	The way zero is displayed on scaling axes
`unitScaleFactor`	*integer*	The scaling axis unit scale exponent to be able to display very small or very large tick values in a small area. Value range is [-24, 24].
`roundMinMaxValues`	*Boolean*	Specifies whether limit values of the scaling axis are rounded to the nearest tick value

Elements	Description
`<minValue>`	Defines the way the maximal value of the scaling axis is managed **Attribute**: `type` (`Auto|Fixed`). Sets a formula if `type` is `Fixed`.
`<maxValue>`	Defines the way the maximal value of the scaling axis is managed **Attribute**: `type` (`Auto|Fixed`). Sets a formula if `type` is `Fixed`.

Example

```
<scaling mode="Linear" unitScaleFactor="0" roundMinMaxValues="false">
    <minValue type="Auto"/>
    <maxValue type="Auto"/>
```

6.3 Date and Time Formats

The RESTful Web Service SDKs support `dateTime` data of the following UTC format:

```
yyyy-mm-ddThh:mm:ss(.s+)(zzzzzz|Z)
```

Where:

Character	Description
`yyyy`	A four digit that represents the year
`-`	Separators between parts of the date portion

Character	Description
First `mm`	A two-digit numeral that represents the month
`dd`	A two-digit numeral that represents the day
`T`	A separator indicating that time-of-day follows
`hh`	A two-digit numeral that represents the hour
`:`	A separator between parts of the time-of-day portion
Second `mm`	A two-digit numeral that represents the minute
`ss`	A two-integer-digit numeral that represents the whole seconds
`.s+`	The fractional seconds preceded by a dot separator
`zzzzzz`	`zzzzzz` represents the timezone according to the W3C recommendation. `zzzzzz` is of the form `+|-hh:mm`.
`Z`	The zero-length duration timezone, which is the UTC canonical representation

Example

`2002-10-10T12:00:00.000-05:00` (noon on 10 October 2002, Central Daylight Savings Time as well as Eastern Standard Time in the U.S.) is equivalent to `2002-10-10T17:00:00.000Z`.

Related Information

http://www.w3.org/TR/xmlschema-2/#dateTime

6.4 Queries

A query specification is an XML document that describes thoroughly the query to be run using the BI Semantic Layer RESTful Web Service SDK. The query specification is based on a query model designed to standardize the data extraction from data providers by products that use the concepts of the Semantic Layer.

This query model allows you to define any type of SQL query to extract and return data from any data source, such as relational databases, OLAP data providers, XML files or web services.

A query specification supports the following components of a query:

- Query options
- Result objects
- Sort objects
- Query filters

The query model also allows you to define advanced queries such as combined queries.

Query Specification Body Schema [page 106]

Query Options [page 107]

Result Objects [page 108]

Sort Objects [page 109]

Query Filters [page 110]

Combined Queries [page 127]

The OData Query Service [page 129]

6.4.1 Query Specification Body Schema

Body Schema (XML)

(`POST /queries` and `GET /queries/<queryID>`)

```
<query xmlns="http://www.sap.com/rws/sl/universe" id="string" dataSourceType="unv|
unx" dataSourceId="integer">
    <querySpecification version="1.0">
        <queryOptions>
               <queryOption>
        <queryData>
            <resultObjects>
                <resultObject>
            <sortObjects>
                <sortObject>
            <filterPart>
                QueryFilters
```

QueryFilters stands for the filter conditions contained by the query. See the next sections for a detailed description of the query specification elements.

Query

Element	Description
`<querySpecification>`	The query specification **Attribute:** `version` (1.0)
`<queryOptions>`	The list of query options
`<queryData>`	The objects used in the query as result, sort and filter objects. There is at least one `<queryData>`

Attribute	Type or Value	Description
`id`	*string*	The query identifier
`dataSourceType`	`unv|unx`	The universe type
`dataSourceId`	*integer*	The universe identifier

Related Information

Query Filters [page 110]
Creating a Query [page 180]

6.4.2 Query Options

You can specify in the query some of the options that you can set in the universe design tool for UNV universes or the information design tool for UNX universes.

```
<queryOptions>
    <queryOption name="string" activated="Boolean" value="string"/>
```

> → Remember
>
> An option that is not present explicitly in the specification is not validated.

Attribute	Type or Value	Description
`name`	*string* (values in table below)	The option name
`activated`	*Boolean*	The option does not apply if `activated` is `false`. Is optional. If not present, the option is activated.
`value`	*string*	The option value

name Attribute Value	Description
`duplicatedRows`	If `true`, the query returns all related rows, even if there are duplicated rows.
`maxRetrievalTimeInSeconds`	It defines the maximum time that a query can run before the query is stopped.
`maxRowsRetrieved`	If `true`, the query returns all the possible rows, but only displays the first n rows,where n is the maximum number of rows set for this option. If the user only needs a certain amount of data, you can set this option to restrict the number of rows of data displayed in reports.
`samplingResultSetSize`	It defines the maximum number of rows that a query returns as a sample.

name Attribute Value	Description
`samplingResultSetFixed`	It defines the type of sampling used. Option values are `true` for fixed sampling and `false` for random.

> → Remember
>
> `samplingResultSetSize` can be set without `samplingResultSetFixed`. In that case, the type of sampling is random.

Example

The following snippet of a query specification for a UNX universe shows the query options.

```
<query xmlns="http://www.sap.com/rws/sl/universe" id="589789982204141561
dataSourceType="unx" dataSourceId="5808">
    <querySpecification version="1.0">
        <queryOptions>
            <queryOption name="duplicatedRows" value="true"/>
            <queryOption name="maxRetrievalTimeInSeconds" activated="false"
value="600"/>
            <queryOption name="maxRowsRetrieved" activated="true" value="5000"/>
            <queryOption name="samplingResultSetSize" activated="false"
value="200">
            <queryOption name="samplingResultSetFixed" activated="false"
value="false">
        </queryOptions>
        ...
    </querySpecification>
</query>
```

6.4.3 Result Objects

The result objects compose the query.

```
<resultObjects>
    <resultObject id="string" path="string"/>
```

Attribute	Type or Value	Description
`id`	*string*	The object identifier
`path`	*string*	The full path of the object in the universe. Optional.

> → Remember
>
> The object order in `<resultObjects>` is important, because it reflects the object order in the SQL query. For example, the following query:
>
> ```
> <queryData>
>  <resultObjects>
> ```

```
        <resultObject id=A .../>
        <resultObject id=B .../>
    </resultObjects>
...
```

gives different results from the following one:

```
<queryData>
    <resultObjects>
        <resultObject id=B .../>
        <resultObject id=A .../>
    </resultObjects>
...
```

Example

The following snippet of a query specification shows the result objects of the query for a UNX universe.

```
<query xmlns="http://www.sap.com/rws/sl/universe" id="5897899822041415615"
dataSourceType="unx" dataSourceId="5808">
    <querySpecification version="1.0">
        <queryOptions>
            ...
        </queryOptions>
        <queryData>
            <resultObjects>
                <resultObject path="Product|folder\Product|dimension"
                    id="_IB8eGbIhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Inventory|folder\Stock Level|measure"
                    id="_IB8eFrIhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Product|folder\Minimum Stock|measure"
                     id="_IB8eHrIhEeCk0Ylv-tlF2Q"/>
            </resultObjects>
        </queryData>
    </querySpecification>
</query>
```

6.4.4 Sort Objects

The sort objects are used to sort the query result. They can be dimensions, attributes or measures. Result objects can be used as sort objects. This mainly depends on the query capabilities of the universe.

```
<sortObjects>
    <sortObject id="string" path="string" sortType="Ascending|Descending"/>
```

Attribute	Type or Value	Description
`id`	*string*	The object identifier, defined when retrieving the data source metadata
`path`	*string*	The full path of the object in the universe. Optional.

Attribute	Type or Value	Description
sortType	Ascending| Descending	The sort type

Example

The following snippet of a query specification shows the sort objects used in the query for a UNX universe. Two of them are also defined as result objects.

```
<query xmlns="http://www.sap.com/rws/sl/universe" id="9168123992538053733"
dataSourceType="unx" dataSourceId="5808">
    <querySpecification version="1.0">
        <queryOptions>
            ...
        <queryOptions>
        <queryData>
            <resultObjects>
                <resultObject path="Product|folder\Category|dimension"
                    id="_IB8eG7IhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Product|folder\Product|dimension"
                    id="_IB8eGbIhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Time|folder\Calendar|folder\Calendar Year
Month|dimension"
                    id="_IBo8G7IhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Inventory|folder\Stock Level|measure"
                    id="_IB8eFrIhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Product|folder\Minimum Stock|measure"
                    id="_IB8eHrIhEeCk0Ylv-tlF2Q"/>
            </resultObjects>
            <sortObjects>
                <sortObject sortType="Ascending" path="Product|folder\Category|
dimension"
                    id="_IB8eG7IhEeCk0Ylv-tlF2Q"/>
                <sortObject sortType="Ascending" path="Product|folder\Product|
dimension"
                    id="_IB8eGbIhEeCk0Ylv-tlF2Q"/>
                <sortObject sortType="Descending" path="Time|folder\Calendar|
folder\Calendar Week
                    |dimension" id="_IBo8HLIhEeCk0Ylv-tlF2Q"/>
            </sortObjects>
        </queryData>
    </querySpecification>
</query>
```

6.4.5 Query Filters

The query filters define the filtering conditions for query results.

The query specification model defines the following types of filters:

- Predefined filters
- Custom filters

The following custom filters are supported:

- Comparison filters based on constant values
- Comparison filters based on objects
- Ranking filters
- Subquery filters
- Combined filters, which are made of multiple comparison filters

Predefined Filters [page 111]

Custom Filters - Comparison Filters with Zero, One, or Two Constant Operands [page 112]

Custom Filters - Comparison Filters for List of Values Operands [page 116]

Custom Filters - Comparison Filters with Parameters [page 117]

Custom Filters - Object Comparison Filters [page 119]

Custom Filters - Ranking Filters [page 121]

Custom Subquery Filters [page 124]

Combined Custom Filters [page 126]

6.4.5.1 Predefined Filters

```
<filterPart>
    <predefinedFilter id="string" path="string"/>
```

Attribute	Type or Value	Description
id	*string*	The object identifier, as defined when retrieving the data source metadata
path	*string*	The full path of the object in the universe

Example

The following snippet of a query specification shows a predefined filter in the query.

```
<query xmlns="http://www.sap.com/rws/sl/universe" id="5897899822041415615"
dataSourceType="unx" dataSourceId="5808">
    <querySpecification version="1.0">
        <queryOptions>
            ...
        </queryOptions>
        <queryData>
            <resultObjects>
                <resultObject path="Product|folder\Product|dimension"
id="_IB8eGbIhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Inventory|folder\Stock Level|measure"
id="_IB8eFrIhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Product|folder\Minimum Stock|measure"
id="_IB8eHrIhEeCk0Ylv-tlF2Q"/>
            </resultObjects>
            <filterPart>
```

```
                 <predefinedFilter path="Inventory|folder\Stock Below Minimum|
filter" id="_V5_GkLR_EeCZotjuycya1A"/>
            </filterPart>
        </queryData>
    </querySpecification>
</query>
```

6.4.5.2 Custom Filters - Comparison Filters with Zero, One, or Two Constant Operands

```
<filterPart>
    <comparisonFilter id="string" path="string" operator="string">
        <constantOperand searchPattern="Boolean">
            <value>
                <caption type="String|Numeric|Date">
```

For each `<comparisonFilter>`, you can add up to two `<constantOperand>`. Each operand contains a `<value>`/`<caption>` that represents the constant value used as right operand.

> **Note**
>
> The following format is still supported:
>
> ```
> <filterPart>
> <comparisonFilter id="string" path="string" operator="string">
> <constantOperand searchPattern="Boolean">
> <answerValue dataType="String|Numeric|Date">
> ```

Comparison Filter

Attribute	Type or Value	Description
`id`	*string*	The identifier of the object used as left operand
`path`	*string*	The full path of the object in the universe
`operator`	*string* (values in table below)	The operator

Operator	operator Attribute Value	Number of Right Operands
Is Null	`IsNull`	Zero
Is Not Null	`IsNotNull`	Zero
Equal To (=)	`EqualTo`	One
Not Equal To (<>)	`NotEqualTo`	One

Operator	operator Attribute Value	Number of Right Operands
Less Than (<)	`LessThan`	One
Greater Than (>)	`GreaterThan`	One
Less Than or Equal To (<=)	`LessThanOrEqualTo`	One
Greater Than or Equal to (>=)	`GreaterThanOrEqualTo`	One
Like	`Like`	One
Not Like	`NotLike`	One
Except	`Except`	One
Between	`Between`	Two
Not Between	`NotBetween`	Two
Both	`Both`	Two

Constant Operand

Attribute	Type or Value	Description
`searchPattern`	*Boolean*	Specifies how the value of `<value>`/`<caption>` or `<answerValue>` is parsed without knowing the escape and mask characters of the underlying database. Can be used with the Like operator only. If set to `true`, the value is parsed: • The * character is replaced with the multicharacter mask of the database in the resulting SQL. • The ? character is replaced with the single-character mask of the database in the reuslting SQL. • If the pattern contains \? or *, then ? or * are escaped. Strings containing ? or * are actually found by the operator. • The _ and % wildcard characters are escaped. If set to `false`, the value is not parsed. Default is `false`.

Answer Value

Attribute	Type or Value	Description
`dataType`	`String|Numeric|Date`	The data type of the value used as filter

Caption

Attribute	Type or Value	Description
`type`	`String|Numeric|Date`	The data type of the value used as filter

Example: Is Null Operator (No Constant Operand)

The following snippet of a query specification shows an `IsNull` comparison filter with no constant operand in the query.

```
<query dataSourceId="5909" dataSourceType="unx" xmlns="http://www.sap.com/rws/sl/
universe">
    <querySpecification version="1.0">
        <queryData>
            <resultObjects>
                <resultObject id="_IBo8M7IhEeCk0Ylv-tlF2Q" path="Customer|
folder\Geography|folder\Continent|dimension"/>
                <resultObject id="_IBo8NrIhEeCk0Ylv-tlF2Q" path="Customer|
folder\Geography|folder\Country|dimension"/>
            </resultObjects>
            <filterPart>
                <comparisonFilter id="_IB8eVbIhEeCk0Ylv-tlF2Q" path="Sales
Orders|folder\Number of Orders|measure"
                    operator="IsNull"/>
            </filterPart>
        </queryData>
    </querySpecification>
</query>
```

Example: Equal To Operator (One Constant Operand)

The following snippet of a query specification shows an `EqualTo` comparison filter with one constant operand in the query.

```
<query xmlns="http://www.sap.com/rws/sl/universe" id="5897899822041415615"
dataSourceType="unx" dataSourceId="5808">
    <querySpecification version="1.0">
        <queryOptions>
            ...
```

```
        </queryOptions>
        <queryData>
            <resultObjects>
                <resultObject path="Customer|folder\Geography|folder\Continent|
dimension" id="_IBo8M7IhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Customer|folder\Geography|folder\Country|
dimension" id="_IBo8NrIhEeCk0Ylv-tlF2Q"/>
                <resultObject path="SalesOrders|folder\Number of Orders|measure"
id="_IB8eVbIhEeCk0Ylv-tlF2Q"/>
            </resultObjects>
            <filterPart>
                <comparisonFilter operator="EqualTo" path="Time|folder\Calendar|
folder\CalendarYear|dimension"
                    id="_IBo8FLIhEeCk0Ylv-tlF2Q">
                    <constantOperand>
                        <value>
                            <caption type="String">2011</caption>
                        </value>
                    </constantOperand>
                </comparisonFilter>
            </filterPart>
        </queryData>
    </querySpecification>
</query>
```

Example: Between Operator (Two Constant Operands)

The following snippet of a query specification shows a `Between` comparison filter with two constant operands in the query.

```
<query dataSourceId="5909" dataSourceType="unx" xmlns="http://www.sap.com/rws/sl/
universe">
    <querySpecification version="1.0">
        <queryData>
            <resultObjects>
                <resultObject id="_IBo8M7IhEeCk0Ylv-tlF2Q" path="Customer|
folder\Geography|folder\Continent|dimension"/>
                <resultObject id="_IBo8NrIhEeCk0Ylv-tlF2Q" path="Customer|
folder\Geography|folder\Country|dimension"/>
                <resultObject id="_IB8eVbIhEeCk0Ylv-tlF2Q" path="Sales Orders|
folder\Number of Orders|measure"/>
            </resultObjects>
            <filterPart>
                <comparisonFilter id="_IB8eVbIhEeCk0Ylv-tlF2Q" path="Sales
Orders|folder\Number of Orders|measure"
                    operator="Between">
                    <constantOperand searchPattern="false">
                        <value>
                            <caption type="Numeric">200</caption>
                        </value>
                    </constantOperand>
                    <constantOperand searchPattern="false">
                        <value>
                            <caption type="Numeric">700</caption>
                        </value>
                    </constantOperand>
                </comparisonFilter>
            </filterPart>
        </queryData>
    </querySpecification>
</query>
```

6.4.5.3 Custom Filters - Comparison Filters for List of Values Operands

```
<filterPart>
    <comparisonFilter id="string" path="string" operator="string">
        <constantOperand>
            <value>
                <caption type=""String|Numeric|Date">
                <path>
                    <level>
                        <caption type=""String|Numeric|Date">
```

The operand contains a `<caption>` that represents the constant value from the list of values used as right operand.

The operand contains `<path>/<level>/<caption>` if the list of values is hierarchical. Levels are flattened in the query specification.

Comparison Filter

Attribute	Type or Value	Description
`id`	*string*	The identifier of the object used as left operand
`path`	*string*	The full path of the object in the universe
`operator`	*string* (values in table below)	The operator

Operator	operator Attribute Value	Number of Right Operands
In	`InList`	List
Not In	`NotInList`	List

Caption

Attribute	Type or Value	Description
`type`	`String|Numeric|Date`	The data type of the value used as filter

Example: InList Operator

The following snipped of a query specification shows an `InList` comparison filter with two cities selected from a list of values.

```
<filterPart>
    <comparisonFilter operator="InList" id="_zPwYENK-EeSNS_-8mYpikg" path="Store
Names|folder\Associate_LOV_City|dimension">
        <constantOperand>
            <value>
                <caption type="String">London</caption>
            </value>
            <value>
                <caption type="String">Paris</caption>
            </value>
        </constantOperand>
    </comparisonFilter>
</filterPart>
```

The following snippet of a query specification shows an `InList` comparison filter with one city selected from a hierarchical list of values.

```
<filterPart>
    <comparisonFilter operator="InList" id="_zPwYENK-EeSNS_-8mYpikg" path="Store
Names|folder\Cascading_Associate_LOV_City|dimension">
        <constantOperand>
            <value>
                <caption type="String">London</caption>
                    <path>
                        <level>
                            <caption type="String">England</caption>
                        </level>
                        <level>
                            <caption type="String">London</caption>
                        </level>
                    </path>
            </value>
        </constantOperand>
    </comparisonFilter>
</filterPart>
```

6.4.5.4 Custom Filters - Comparison Filters with Parameters

```
<filterPart>
    <comparisonFilter id="string" path="string" operator="string">
        <parameterOperand>
            <parameter constrained="Boolean" keepLastValues="Boolean"
optional="Boolean" promptWithLov="Boolean">
                <question>
```

The operand contains a parameter and a question that needs to be answered to get the filter values.

Comparison Filter

Attribute	Type or Value	Description
`id`	*string*	The identifier of the object used as left operand
`path`	*string*	The full path of the object in the universe
`operator`	*string* (values in table below)	The operator

Operator	operator Attribute Value	Number of Right Operands
Is Null	`IsNull`	Zero
Is Not Null	`IsNotNull`	Zero
Equal To (=)	`EqualTo`	One
Not Equal To (<>)	`NotEqualTo`	One
Less Than (<)	`LessThan`	One
Greater Than (>)	`GreaterThan`	One
Less Than or Equal To (<=)	`LessThanOrEqualTo`	One
Greater Than or Equal to (>=)	`GreaterThanOrEqualTo`	One
Like	`Like`	One
Not Like	`NotLike`	One
Except	`Except`	One
Between	`Between`	Two
Not Between	`NotBetween`	Two
Both	`Both`	Two

Parameter

Attribute	Type or Value	Description
`constrained`	*Boolean*	Specifes if the user can type a new value for the parameter (`false`) or if the user must select only the values from the associated list of values (`true`).

Attribute	Type or Value	Description
`keepLastValues`	*Boolean*	Indicates whether the parameter keeps the previous answered values. → Remember When working with the BI Semantic Layer RESTful web service, you must implement the storage of the previous answered values, because the web service does not store them.
`optional`	*Boolean*	Specifies if the parameter is optional.
`promptWithLov`	*Boolean*	Specifies if the parameter is associated with a list of values.

Example

The following snippet of a query specification shows an `EqualTo` comparison filter with a prompt that needs to be answered to get the filter values.

```
<filterPart xmlns="http://www.sap.com/rws/sl/universe">
    <comparisonFilter id="_6zk08A-8EeO1RP--CtxScg" operator="EqualTo"
path="Custorder\OrderId">
        <parameterOperand>
            <parameter constrained="true" keepLastValues="true" optional="false"
promptWithLov="true">
                <question>Enter Order ID</question>
            </parameter>
        </parameterOperand>
    </comparisonFilter>
</filterPart>
```

6.4.5.5 Custom Filters - Object Comparison Filters

```
<filterPart>
    <comparisonFilter id="string" path="string" operator="string">
        <objectOperand id="string" path="string">
```

`<objectOperand>` represents the business object used as right operand.

Comparison Filter

Attribute	Type or Value	Description
`id`	*string*	The identifier of the object used as left operand, as defined when retrieving the data source metadata
`path`	*string*	The full path of the object in the universe

Attribute	Type or Value	Description
`operator`	*string* (values in table below)	The operator

Operator	operator Attribute Value	Number of Right Operands
Equal To (=)	`EqualTo`	One object
Not Equal To (<>)	`NotEqualTo`	One object
Less Than (<)	`LessThan`	One object
Greater Than (>)	`GreaterThan`	One object
Less Than or Equal To (<=)	`LessThanOrEqualTo`	One object
Greater Than or Equal to (>=)	`GreaterThanOrEqualTo`	One object

Object Operand

Attribute	Type or Value	Description
`id`	*string*	The identifier of the object used as right operand
`path`	*string*	The full path of the object in the universe

Example

The following snippet of a query specification shows an `EqualTo` comparison filter with one object as operand in the query.

```
<query xmlns="http://www.sap.com/rws/sl/universe" id="5897899822041415615"
dataSourceType="unx" dataSourceId="5808">
    <querySpecification version="1.0">
        <queryOptions>
            ...
        </queryOptions>
        <queryData>
            <resultObjects>
                <resultObject path="Customer|folder\Geography|folder\Continent|
dimension" id="_IBo8M7IhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Customer|folder\Geography|folder\Country|
dimension" id="_IBo8NrIhEeCk0Ylv-tlF2Q"/>
                <resultObject path="SalesOrders|folder\Number of Orders|measure"
id="_IB8eVbIhEeCk0Ylv-tlF2Q"/>
            </resultObjects>
            <filterPart>
                <comparisonFilter path="Customer|folder\Geography|folder\Country|
dimension" operator="EqualTo"
                    id="_IBo8NrIhEeCk0Ylv-tlF2Q">
```

```
                    <objectOperand id="_IBo8JzLhEeCk0Ylv-tlF2Q"
path="Restrictions\Restricted Country"/>
                </comparisonFilter>
            </filterPart>
        </queryData>
    </querySpecification>
</query>
```

6.4.5.6 Custom Filters - Ranking Filters

```
<filterPart>
    <rankingFilter function="Top|Bottom|topPercent|bottomPercent" level="integer">
        <prompt>
        <dimension id="string" path="string"/>
        <basedOnMeasure id="string" path="string"/>
        <rankedByDimensions>
            <rankedByDimension id="string" path="string"/>
        <filterPart>
```

The ranking filter is defined by:

- A ranking order (ascending or descending)
- The question prompted to the end-user if the filter is triggered by a prompt
- The dimension and measure to rank
- The additional dimensions used to rank
- Any additional custom filter it may contain

Ranking Filter

Attribute	Type or Value	Description
`function`	`Top|Bottom|topPercent|bottomPercent`	Specifies if the ranking is ascending (`Bottom`, `bottomPercent`) or descending (`Top`, `topPercent`)
`level`	*integer*	The number of values to rank. Optional. In the case of a filter with prompt, it is the default value of the prompt. This behavior is specific to UNV universes.

Prompt

Element	Type or Value	Description
`<prompt>`	*string*	Specifies the prompt question in the case of a filter with prompt

Dimension

Attribute	Type or Value	Description
id	*string*	The identifier of the dimension object to rank
path	*string*	Optional.The full path of the dimension object

Measure

Attribute	Type or Value	Description
id	*string*	The identifier of the measure object to rank
path	*string*	Optional.The full path of the measure object

Ranked by Dimensions

`<rankedByDimensions>` is optional. You add at least one `<rankedByDimension>` to `<rankedByDimensions>`.

Attribute	Type or Value	Description
id	*string*	The identifier of the additional dimension object used to rank
path	*string*	Optional. The full path of the dimension object

Example: Without Prompt

```
<query dataSourceType="unx" dataSourceId="91800" xmlns="http://
www.sap.com/rws/sl/universe">
    <querySpecification version="1.0">
        <queryData>
            <resultObjects>
                <resultObject path="Dimcustomer|folder\Regionname|dimension"
id="_60xHwQ-8EeO1RP--CtxScg"/>
                <resultObject path="Dimcustomer|folder\Countryname|dimension"
id="_60xHwA-8EeO1RP--CtxScg"/>
                <resultObject path="Dimcustomer|folder\Customer Name|dimension"
id="_7Zkd8A-8EeO1RP--CtxScg"/>
                <resultObject path="Custorderline|folder\Quantity|measure"
id="_60Bg4g-8EeO1RP--CtxScg"/>
            </resultObjects>
            <filterPart>
                <rankingFilter level="3" function="Top">
                    <dimension path="Dimcustomer|folder\Customer Name|dimension"
id="_7Zkd8A-8EeO1RP--CtxScg"/>
                    <basedOnMeasure path="Custorderline|folder\Quantity|measure"
id="_60Bg4g-8EeO1RP--CtxScg"/>
```

```
                </rankingFilter>
            </filterPart>
        </queryData>
    </querySpecification>
</query>
```

Example: With a Prompt

```
<query dataSourceType="unx" dataSourceId="91800" xmlns="http://
www.sap.com/rws/sl/universe">
    <querySpecification version="1.0">
        <queryData>
            <resultObjects>
                <resultObject path="Dimcustomer|folder\Regionname|dimension"
id="_60xHwQ-8EeO1RP--CtxScg"/>
                <resultObject path="Dimcustomer|folder\Countryname|dimension"
id="_60xHwA-8EeO1RP--CtxScg"/>
                <resultObject path="Dimcustomer|folder\Customer Name|dimension"
id="_7Zkd8A-8EeO1RP--CtxScg"/>
                <resultObject path="Custorderline|folder\Quantity|measure"
id="_60Bg4g-8EeO1RP--CtxScg"/>
            </resultObjects>
            <filterPart>
                <rankingFilter level="3" function="Top">
                    <prompt>Enter the ranking level:</prompt>
                    <dimension path="Dimcustomer|folder\Customer Name|dimension"
id="_7Zkd8A-8EeO1RP--CtxScg"/>
                    <basedOnMeasure path="Custorderline|folder\Quantity|measure"
id="_60Bg4g-8EeO1RP--CtxScg"/>
                </rankingFilter>
            </filterPart>
        </queryData>
    </querySpecification>
</query>
```

Example: With Additional Dimensions

```
<query dataSourceType="unx" dataSourceId="91800" xmlns="http://
www.sap.com/rws/sl/universe">
    <querySpecification version="1.0">
        <queryData>
            <resultObjects>
                <resultObject path="Dimcustomer|folder\Regionname|dimension"
id="_60xHwQ-8EeO1RP--CtxScg"/>
                <resultObject path="Dimcustomer|folder\Countryname|dimension"
id="_60xHwA-8EeO1RP--CtxScg"/>
                <resultObject path="Dimcustomer|folder\Customer Name|dimension"
id="_7Zkd8A-8EeO1RP--CtxScg"/>
                <resultObject path="Custorderline|folder\Quantity|measure"
id="_60Bg4g-8EeO1RP--CtxScg"/>
            </resultObjects>
            <filterPart>
                <rankingFilter level="3" function="Top">
                    <prompt>Enter ranking level :</prompt>
                    <dimension path="Dimcustomer|folder\Customer Name|dimension"
id="_7Zkd8A-8EeO1RP--CtxScg"/>
                    <basedOnMeasure path="Custorderline|folder\Quantity|measure"
id="_60Bg4g-8EeO1RP--CtxScg"/>
```

```
                    <rankedByDimensions>
                        <rankedByDimension path="Dimcustomer|folder\Regionname|
dimension" id="_60xHwQ-8EeO1RP--CtxScg"/>
                        <rankedByDimension path="Dimcustomer|folder\Countryname|
dimension" id="_60xHwQ-8EeO1RP--CtxScg"/>
                    </rankedByDimensions>
                </rankingFilter>
            </filterPart>
        </queryData>
    </querySpecification>
</query>
```

Example: With a Predefined Filter

```
<query dataSourceType="unx" dataSourceId="91800" xmlns="http://
www.sap.com/rws/sl/universe">
    <querySpecification version="1.0">
        <queryData>
            <resultObjects>
                <resultObject path="Dimcustomer|folder\Regionname|dimension"
id="_60xHwQ-8EeO1RP--CtxScg"/>
                <resultObject path="Dimcustomer|folder\Countryname|dimension"
id="_60xHwA-8EeO1RP--CtxScg"/>
                <resultObject path="Dimcustomer|folder\Customer Name|dimension"
id="_7Zkd8A-8EeO1RP--CtxScg"/>
                <resultObject path="Custorderline|folder\Quantity|measure"
id="_60Bg4g-8EeO1RP--CtxScg"/>
            </resultObjects>
            <filterPart>
                <rankingFilter level="3" function="Top">
                    <dimension path="Dimcustomer|folder\Customer Name|dimension"
id="_7Zkd8A-8EeO1RP--CtxScg"/>
                    <basedOnMeasure path="Custorderline|folder\Quantity|measure"
id="_60Bg4g-8EeO1RP--CtxScg"/>
                    <filterPart>
                        <predefinedFilter path="Filters|folder\European
Customers|filter" id="_9onvcA-8EeO1RP--CtxScg"/>
                    </filterPart>
                </rankingFilter>
            </filterPart>
        </queryData>
    </querySpecification>
</query>
```

6.4.5.7 Custom Subquery Filters

```
<filterPart>
    <subQueryFilter operator="string" correlationType="None|Any|All">
        <filterObjects>
            <filterObject id="string" path="string"/>
        <queryData>
            <resultObjects>
                <resultObject>
            <sortObjects>
                <sortObject>
            <filterPart>
```

See the *SAP BusinessObjects Web Intelligence User's Guide* to learn the definition and benefits of subqueries.

Subquery Filter

Attribute	Type or Value	Description
`operator`	*string*	The operator
`correctionType`	`None|Any|All`	The correlation type

Operator	operator Attribute Value	Number of Operands	Correlation Supported
Equal To (=)	`EqualTo`	One object	No
Not Equal To (<>)	`NotEqualTo`	One object	Yes
Less Than (<)	`LessThan`	One object	Yes
Greater Than (>)	`GreaterThan`	One object	Yes
Less Than or Equal To (<=)	`LessThanOrEqualTo`	One object	Yes
Greater Than or Equal to (>=)	`GreaterThanOrEqualTo`	One object	Yes
In	`InList`	List	No
Not In	`NotInList`	List	No

Filter Object

Attribute	Type or Value	Description
`id`	*string*	The object identifier used as filter
`path`	*string*	The full path of the object in the universe

Example

The following snippet of a query specification shows a subquery filter with no correlation.

```
<query xmlns="http://www.sap.com/rws/sl/universe" id="5897899822041415615"
dataSourceType="unx" dataSourceId="5808">
    <querySpecification version="1.0">
        <queryOptions>
        ...
       </queryOptions>
       <queryData>
            <resultObjects>
```

```
                <resultObject path="Time|folder\Calendar|folder\Calendar Year|
dimension" id="_IBo8FLIhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Customer|folder\Geography|folder\Continent|
dimension" id="_IBo8M7IhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Customer|folder\Geography|folder\Country|
dimension" id="_IBo8NrIhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Sales Revenue|folder\Net Sales|measure"
id="_IB8eRLIhEeCk0Ylv-tlF2Q"/>
            </resultObjects>
            <filterPart>
                <subQueryFilter operator="EqualTo" correlationType="Any">
                    <filterObjects>
                        <filterObject path="Customer|folder\Geography|
folder\Continent|dimension"
                            id="_IBo8M7IhEeCk0Ylv-tlF2Q"/>
                        <filterObject path="Customer|folder\Geography|
folder\Country|dimension"
                            id="_IBo8NrIhEeCk0Ylv-tlF2Q"/>
                    </filterObjects>
                    <queryData>
                        <resultObjects>
                            <resultObject path="Customer|folder\Geography|
folder\Continent|dimension"
                                id="_IBo8M7IhEeCk0Ylv-tlF2Q"/>
                            <resultObject path="Customer|folder\Geography|
folder\Country|dimension"
                                id="_IBo8NrIhEeCk0Ylv-tlF2Q"/>
                        </resultObjects>
                        <filterPart>
                            <comparisonFilter operator="GreaterThan" path="Sales
Revenue|folder\Net Sales|measure"
                                id="_IB8eRLIhEeCk0Ylv-tlF2Q">
                                <constantOperand>
                                    <value>
                                        <caption type="Numeric">300000000</
caption>
                                    </value>
                                </constantOperand>
                            </comparisonFilter>
                        </filterPart>
                    </queryData>
                </subQueryFilter>
            </filterPart>
        </queryData>
    </querySpecification>
</query>
```

6.4.5.8 Combined Custom Filters

In the query specification model, comparison filters can be combined using the AND and OR logical operators. These operators are defined using the `<and>` and `<or>` XML elements.

Example

The following snippet of a query specification shows the combination AND of two comparison filters.

```
<query xmlns="http://www.sap.com/rws/sl/universe" id="5897899822041415615"
dataSourceType="unx" dataSourceId="5808">
```

```
    <querySpecification version="1.0">
        <queryOptions>
            ...
        </queryOptions>
        <queryData>
            <resultObjects>
                <resultObject path="Customer|folder\Geography|folder\Country|
dimension" id="_IBo8NrIhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Customer|folder\Customer|dimension"
id="_IBo8OrIhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Customer Satisfaction|folder\Satisfaction
Index|measure" id="_IBytE7IhEeCk0Ylv-tlF2Q"/>
            </resultObjects>
            <filterPart>
                <and>
                    <comparisonFilter path="Time|folder\Calendar|folder\Calendar
Year Month|dimension" operator="EqualTo"
                        id="_IBo8G7IhEeCk0Ylv-tlF2Q">
                        <constantOperand>
                            <value>
                                <caption type="String">CY2011-M10</caption>
                            </value>
                        </constantOperand>
                    </comparisonFilter>
                    <comparisonFilter path="Customer|folder\Geography|
folder\Continent|dimension" operator="EqualTo"
                        id="_IBo8M7IhEeCk0Ylv-tlF2Q">
                        <constantOperand>
                            <value>
                                <caption type="String">Europe</caption>
                            </value>
                         </constantOperand>
                    </comparisonFilter>
                </and>
            </filterPart>
        </queryData>
    </querySpecification>
</query>
```

6.4.6 Combined Queries

The BI Semantic Layer RESTful Web Service SDK allows a client tool to build an XML query specification that contains multiple queries combined using operators. Only one query result is returned.

The following operators are supported:

Operator	XML Element
UNION	`union`
MINUS	`minus`
INTERSECT	`intersect`

See the *SAP BusinessObjects Web Intelligence User Guide* for more information on these operators.

Example

The following snippet of a query specification shows two queries combined with the MINUS operator.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<query xmlns="http://www.sap.com/rws/sl/universe" id="5897899822041415615"
dataSourceType="unx" dataSourceId="5808">
    <querySpecification version="1.0">
        <queryOptions>
            ...
        </queryOptions>
        <minus>
            <queryData>
                <resultObjects>
                    <resultObject path="Customer|folder\Customer|dimension"
id="_IBo8OrIhEeCk0Ylv-tlF2Q"/>
                    <resultObject path="Customer|folder\Geography|folder\Country|
dimension" id="_IBo8NrIhEeCk0Ylv-tlF2Q"/>
                    <resultObject path="Customer Satisfaction|
folder\Satisfaction Index|measure"
                        id="_IBytE7IhEeCk0Ylv-tlF2Q"/>
                    <resultObject path="Customer Satisfaction|folder\Achievement
rate|measure"
                        id="_-OwM8LOlEeCMD5vI8SKUZA"/>
                </resultObjects>
                <sortObjects>
                    <sortObject path="Customer Satisfaction|folder\Achievement
rate|measure" sortType="Ascending"
                        id="_-OwM8LOlEeCMD5vI8SKUZA"/>
                </sortObjects>
                <filterPart>
                     <comparisonFilter path="Customer|folder\Geography|
folder\Country|dimension" operator="InList"
                         id="_IBo8NrIhEeCk0Ylv-tlF2Q">
                         <constantOperand>
                             <value>
                                 <caption type="String">France</caption>
                             </value>
                             <value>
                                 <caption type="String">Germany</caption>
                             </value>
                         </constantOperand>
                     </comparisonFilter>
                </filterPart>
            </queryData>
            <queryData>
                <resultObjects>
                    <resultObject path="Customer|folder\Customer|dimension"
id="_IBo8OrIhEeCk0Ylv-tlF2Q"/>
                    <resultObject path="Customer|folder\Geography|folder\Country|
dimension" id="_IBo8NrIhEeCk0Ylv-tlF2Q"/>
                    <resultObject path="Customer Satisfaction|
folder\Satisfaction Index|measure"
                        id="_IBytE7IhEeCk0Ylv-tlF2Q"/>
                    <resultObject path="Customer Satisfaction|folder\Achievement
rate|measure"
                        id="_-OwM8LOlEeCMD5vI8SKUZA"/>
                </resultObjects>
                <filterPart>
                    <comparisonFilter operator="Between" path="Customer
Satisfaction|folder\Achievement rate|measure"
                        id="_-OwM8LOlEeCMD5vI8SKUZA">
                        <constantOperand>
                            <value>
                                 <caption type="Numeric">0.85</caption>
                            </value>
```

```
                        </constantOperand>
                        <constantOperand>
                            <value>
                                 <caption type="Numeric">1</caption>
                            </value>
                        </constantOperand>
                    </comparisonFilter>
                </filterPart>
            </queryData>
        </minus>
    </querySpecification>
</query>
```

6.4.7 The OData Query Service

The BI Semantic Layer RESTful Web Service SDK allows client tools to use the Open Data (OData) protocol to perform the following tasks:

- Running a query statement
- Retrieving the query results
- Exposing the query results as OData flows

The OData query service exposes the query results according to a schema based on the Entity Data Model (EDM). The schema is described with the help of CSDL. CSDL is an XML format that describes the structure and semantics of Entity Data Model schemas. To learn more about the concepts that rule an entity data model, see the CSDL specification .

The OData query service exposes the results of a query as one flow. It returns multiple flows in the following cases:

- If the query specification defines combined queries
- If the query uses unlinked objects

The schema defines the metadata used to build the query results. It is represented as a file of the EDMX format.

The query is running when the OData service is called. The OData service does not support POST HTTP requests.

Mapping Query Data [page 129]

Naming the Properties [page 131]

Mapping Query Data Types [page 131]

6.4.7.1 Mapping Query Data

The EDM schema used by the OData service maps the following EDM elements:

- The EntityContainer contains EntitySets.
- EntitySets are mapped to query flows. An EntitySet is defined by an EntityType.
- An EntityType consists of a series of Properties. A Property is mapped to a result object defined in the Query Specification and used in a query flow.

The following table describes the attributes defined for a Property. Attributes prefixed with "sap" are specific to SAP BusinessObjects universes.

Attribute	Description
`Name`	Name used by the OData service to identify the object.
`Type`	Result type.
`sap:label`	Original name of the object in the universe.
`sap:objectKey`	Object ID in the universe.
`sap:qualification`	Object type. Possible values are `Dimension`, `Measure`, and `Attribute`.
`sap:projectionFunction`	Aggregation function applied to an object of type `Measure`. Possible values are: `Average`, `Count`, `First`, `Last`, `Max`, `Min`, `Sum`, `Delegated`, and `None.`

Example

The following `<Property>` element defines an object used in the query:

```
<Property Name="Year" Type="Edm.String" Nullable="true" Unicode="true"
sap:objectKey="OBJ_49" sap:qualification="Dimension"
    sap:label="Year"/>
```

Incomplete Result Set

The `<Schema>` XML element has a `sap:isPartial` attribute that indicates whether the result set of the query is complete.

`sap:isPartial` is set to `false` if the result set to return is greater than the value of the "Max rows retrieved" option or if the the execution query time takes more time than the maximum timeout defined in the "Max retrieval time" option. It is set to `true` otherwise.

Related Information

Query Options [page 107]
Getting the OData Flow Metadata [page 207]

6.4.7.2 Naming the Properties

When generating the XML flow, the object names are automatically converted into Property names. The original object name is stored in the `sap:label` attribute of the Property.

In the EDM schema, the value of the `Name` attribute of a Property must adhere to the following rules:

- It cannot start with a number or punctuation character.
- It cannot contain spaces.
- It cannot contain colons (`:`), periods (`.`), and `@` characters.
- Any name can be used, no words are reserved.
- It is not recommended to start with `xml`.

If the generated name is an empty string, then the Property name is set to `col`.

Generated names must be unique. If they appear to be identical after conversion, the Property name is suffixed with `_<increment>`, where `<increment>` starts with `1`.

Example: Property Names and sap:label Values

These examples illustrate the naming rules for `Name` and the corresponding, original object names set in `sap:label`.

```
<Property Name="Year" ... sap:label="Year"/>
<Property Name="Reservation_Year" ... sap:label="Reservation Year"/>
<Property Name="My_Name" ... sap:label="My_Name"/>
<Property Name="My_Name_1" ... sap:label="My Name"/>
<Property Name="inval_d3_name_" ... sap:label="inval!d3name$"/>
<Property Name="col" ... sap:label="$"/>
<Property Name="col_1" ... sap:label="!"/>
```

6.4.7.3 Mapping Query Data Types

The types of the objects used in the query are mapped to the following OData types:

OData Data Types	BI Semantic Layer Data Types
Edm.Binary	BLOB
Edm.Boolean	BOOLEAN
Edm.DateTime	CALENDAR_DATE, DATE, DATE_TIME
Edm.Decimal	CURRENCY, FLOAT
Edm.Double	DOUBLE

OData Data Types	BI Semantic Layer Data Types
Edm.Int32	INT
Edm.Null	NULL
Edm.String	STRING, UNKNOWN
Edm.Time	TIME, TIMESTAMP

6.5 Parameters

The term "parameter" refers to the element defined in the request body manipulated by the web service. It represents contexts, @Prompts, and object parameters.

SDK	Functionality
BI Semantic Layer RESTful Web Service SDK	• Resolve contexts in universes • Answer @Prompts in universes • Answer object parameters defined in data foundations and business layers, and used as filters or in @Prompts.
Web Intelligence RESTful Web Service SDK	• Resolve contexts of queries based on universes • Identify and fill in prompts with values to refresh documents. Prompts may come from @Prompts in universes, object parameters defined in data foundations and business layers, SAP BW parameters or be defined in the query.

Supported parameters and lists of values include:

- Index-aware parameters
- Optional and non optional parameters
- Cascading parameters
- Index-aware lists of values
- Lists of values with multiple columns for UNX universes only
- Hierarchical lists of values for UNX universes and BEx queries
- Object parameters

See the *Information Design Tool User Guide* for more information on contexts, @Prompts, and object parameters.

Parameter Response Body Schemas [page 133]

Lists of Values [page 136]

Default Values [page 140]

Previous Values [page 141]

Incompatible Contexts [page 141]

Answer Request Body Schemas [page 142]

6.5.1 Parameter Response Body Schemas

Response Body Schemas (XML)

(`GET` and `PUT .../parameters`)

This schema contains the parameters and their properties, such as the expected answers.

> **Note**
>
> See examples in Getting the List of Parameters [page 184] (BI Semantic Layer RESTful Web Service SDK) and Getting the Refresh Parameters [page 619] (Web Intelligence RESTful Web Service SDK).

```
<parameters>
    <parameter type="context|prompt|sapVariable" optional="Boolean" dpId="string"
dpLinks="string">
        <id>
        <technicalName>
        <name>
        <description>
        <answer type="Text|Numeric|Date|Unknown" constrained="Boolean"
keyType="Text|Numeric|Date|DateTime|Unknown">
            <info cardinality="Single|Multiple|Interval" keepLastValues="Boolean">
                LOV
                [DefaultValues|PreviousValues]
                ContextIncompatibilities
            Values
```

Parameter

Element	Type or Value	Description
`<id>`	*integer*	Parameter identifier. The web service generates it automatically.
`<technicalName>`	*string*	Optional technical name of the parameter
`<name>`	*string*	Prompt question in the user locale
`<description>`	*string*	Optional description of the parameter
`<answer>`	N/A	The expected answer

Attribute	Type or Value	Description
`type`	`context|prompt|sapVariable`	The parameter type (context, @Prompt, object parameter, or SAP variable)

Attribute	Type or Value	Description
`optional`	*Boolean*	Specifies whether the parameter is optional (`false` if `type="context"`)
`dpId`	*string*	A mandatory attribute that defines the data provider identifier used for this parameter. Only used in the Web Intelligence RESTful Web Service SDK.
`dpLinks`	*string*	An optional attribute that specifies the parameter of type `prompt` is used in several queries (data providers).

Answer

Element	Description
`<info>`	Contains: • The list of values associated with the parameter (*LOV*) • The *DefaultValues* or *PreviousValues* • The incompatible context values (*ContextIncompatibilities*) See the next sections for more information.
Values	The values associated with the parameter. In the `GET .../parameters` call result, they are either *DefaultValues* or *PreviousValues*. → Remember *PreviousValues* are only returned by the Web Intelligence RESTful Web Service SDK.

Attribute	Type or Value	Description
`type`	`Text|Numeric|Date|DateTime|Unknown`	The answer type. The `unknown` type is not supposed to be returned.
`constrained`	*Boolean*	Defines if the user can type a new value for the parameter (`false`) or if the user must select only the values from the associated list of values (`true`).
`keyType`	`Text|Numeric|Date|DateTime|Unknown`	The data type of the key column when the parameter expects indexed answers. The `unknown` type does not display.

Info

Attribute	Type or Value	Description

`cardinality`	`Single`|`Multiple`|
	`Interval`|
	`Intervals`|
	`Condition`|
	`Conditions`
		The number and type of values expected by the parameter are:
		1. Single: One value
		`<answer>`
		`  <values>`
		`  <value>...</value>`
		`  </values>`
		`</answer>`
		2. Multiple: One or several value(s)
		`<answer>`
		`  <values>`
		`  <value>...</value>`
		`  ...`
		`  </values>`
		`</answer>`
		3. Interval: One interval
		`<answer>`
		`  <values>`
		`  <value>...</value>`
		`  <value>...</value>`
		`  </values>`
		`</answer>`
		4. Intervals: One or several interval(s)
		`<answer>`
		`  <values>`
		`  <condition operator="Between">`
		`  <value>...</value>`
		`  <value>...</value>`
		`  </condition>`
		`  ...`
		`  </values>`
		`</answer>`
		5. Condition: One condition
		`<answer>`
		`  <values>`
		`  <condition operator="operator">`
		`  <value>...</value>`
		`  ...`
		`  </condition>`
		`  </values>`
		`</answer>`
		6. Conditions: One or several condition(s)
		`<answer>`
		`  <values>`
		`  <condition operator="operator">`
		`  <value>...</value>`
		`  ...`

Attribute	Type or Value	Description

		`</condition>`
		`...`
		`</values>`
		`</answer>`
		
		Valid operator values for a condition are listed below:
		• NotEqual
		• Greater
		• GreaterOrEqual
		• Less
		• LessOrEqual
		• InList
		• NotInList
		• Between
		• NotBetween
		• IsNull
		• IsNotNull

`keepLastValues`	*Boolean*	Indicates whether the parameter of type `prompt` keeps the previous answered values. → Remember When working with the BI Semantic Layer RESTful web service, you must implement the storage of the previous answered values, because the web service does not store them. When working with the Web Intelligence RESTful web service, the values are kept in the Web Intelligence document.

Related Information

Lists of Values [page 136]
Default Values [page 140]
Previous Values [page 141]
Incompatible Contexts [page 141]

6.5.2 Lists of Values

LOV represents the list of values associated with the parameter. It describes the possible values of the answer.

The `<lov>` element defines a list of values itself.

```
<lov hierarchical="Boolean"
```

```
      partial="Boolean"
      refreshable="Boolean"
      searchable="Boolean"
      mandatorySearch="Boolean"
      path="[[0|1|2,\ second_level,\ index],[0|1|2,\ third_level,\ index],[...]]"
      nodeSelection="Any|Leaf">
```

List of Values

Attribute	Type or Value	Description
`hierarchical`	*Boolean*	Specifies whether the list of values associated with the parameter is hierarchical.
`partial`	*Boolean*	Specifies whether the entire list of values is displayed. The size of the list of values depends on server settings or universe query limit.
`refreshable`	*Boolean*	Specifies whether the list of values can be refreshed. This attribute can be used in a user interface to allow the refresh of the list of values.
`searchable`	*Boolean*	Specifies whether values of the list of values can be searched. This attribute can be used in a user interface to allow the search on a list of values.
`mandatorySearch`	*Boolean*	Specifies whether values of the list of values are restricted to those filtered through a search pattern. This attribute reflects the *Force users to filter values before use* option of the default list of values. If `mandatorySearch="true"`, no value or interval is returned unless a search pattern `<search>` is provided in `<query>`. See Answer Request Body Schemas [page 142]. This is an optional attribute that you can use only when `searchable="true"`.
`path`	`[0|1|2,\ level,\ index]`	Optional. In the case of hierarchical parameters, specifies the types, values, and indexes of the levels of the hierarchy, starting with the second level. This attribute appears in the `lov` element of the response of the `PUT .../parameters` call. The syntax `[0|1|2,\ level,\ index]` describes the data type, value, and index of a level. Indexes are mentioned if the hierarchy has indexed levels. • `0` represents a string • `1` represents a date • `2` represents a number See example in Example - Responding to a Hierarchical Parameter [page 196].
`nodeSelection`	`Any|Leaf`	Optional. In the case of hierarchical parameters, specifies if only values as leaves (`Leaf`) or values as nodes or leaves in the hierarchy (`Any`) can be selected.

One Column

LOV is made of the following elements if the list of values contains one column:

```
<lov>
    <id>
    <updated>
    [Intervals | Values]
     Columns
```

> **Note**
>
> *Columns* does not appear if the parameter is of type context.

Element	Type or Value	Description
`<id>`	*string*	The list of values identifier
`<updated>`	*DateTime*	The date of last update of the list of values

Intervals	N/A	The element block that describes values as intervals when the number of values is too large. The first interval is returned by default. The default number of values in an interval is 50.
		
		Values of a `context` parameter can have a description.
		
		`<intervals>`
		`  <interval id="integer">`
		`  <value id="integer" description="string" final="Boolean">`

Values	N/A	The element block that describes the possible values of the list of values. Values of a `context` parameter can have a description.
		
		`<values>`
		`  <value id="integer" description="string" final="Boolean">`

Columns	N/A	The element block that describes the column to be mapped to the list of values. `<column>` defines the column name. Its `type` attribute is mandatory.
		
		`<columns mappingID="integer">`
		`  <column id="integer" type="String|Date|Numeric">`

Multiple Columns

LOV is made of the following elements if the list of values contains multiple columns:

```
<lov>
    <id>
    <updated>
    [Intervals_MC | Values_MC]
```

Columns

> **Note**
>
> *Columns* does not appear if the parameter is of type context.

Element	Type or Value	Description
`<id>`	*string*	The list of values identifier
`<updated>`	*DateTime*	The date of last update of the list of values

Intervals_MC	N/A	The element block that describes values as intervals when the number of values is too large. The first interval is returned by default. An interval can contain 50 values.
		`<intervals>`
		`  <interval id="integer">`
		`  <cvalue id="integer" final="Boolean">`
		`  <column id="integer">`

Values_MC	N/A	The element block that describes the possible values of the list of values. `<cvalue>` defines the values of multiple columns. `<column>` defines the column value.
		`<cvalues>`
		`  <cvalue id="integer" final="Boolean">`
		`  <column id="integer">`

Columns	N/A	The element block that describes the columns to be mapped to the list of values. `<column>` defines the column name. Its `type` attribute is mandatory. `mappingID` defines the identifier of the column, of which the value is used as reference to answer the parameter.
		`<columns mappingID="integer">`
		`  <column id="integer" type="String|Date|`
		`Numeric">`

Cascading Parameters

In the case of cascading parameters, the value given to a parameter depends on the answer to a previous parameter.

LOV is made of the following elements:

```
<lov>
    <id>
    <parameters>
        <id>
        <id>
        ...
```

Under `<parameters>`, each `<id>` corresponds to the identifier of a parameter, on which depend the values of the current parameter.

6.5.3 Default Values

DefaultValues represents the default values that the parameter can accept.

> **Note**
>
> If the parameter type is context, then there is no default value, and *DefaultValues* is not required.

One Column

DefaultValues is structured as follows if the parameter accepts values of only one column:

```
<values>
    <value id="string" path="[[0|1|2,\ second_level,\ index],[0|1|2,\
third_level,\ index],[...]]">
    ...
```

The following table describes the `<value>` attributes:

Attribute	Type or Value	Description
`id`	*string*	The value identifier. It is used if the parameter is of context type or if the values are indexed either from the SAP system or from index awareness.
`path`	`[0|1|2,\ level,\ index]`	Optional. In the case of hierarchical parameters, specifies the types, values, and indexes of the levels of the hierarchy, starting with the second level. The syntax `[0|1|2,\ level,\ index]` describes the data type, value, and index of a level. Indexes are mentioned if the hierarchy has indexed levels. • `0` represents a string • `1` represents a date • `2` represents a number

Multiple Columns

DefaultValues is structured as follows if the parameter accepts values of multiple columns:

```
<cvalues>
    <cvalue id="string" final="Boolean">
        <column id="integer" type="String|Date|Numeric">
        ...
```

`<cvalue>` defines the values of multiple columns. The following table describes its attributes:

Attribute	Type or Value	Description
`id`	*string*	The value identifier. It is used if the values are indexed either from the SAP system or from index awareness.
`final`	*Boolean*	Attribute specific to hierarchical parameters that defines if the parameter is used to answer a value for a node (`final="false"`) or a leaf (`final="true"`) in the hierarchy

`<column>` defines the value mapped to a column. The following table describes its attributes:

Attribute	Type or Value	Description
`id`	*integer*	The column identifier. The web service generates it automatically.
`type`	`String|Date|Numeric`	The column type. Is optional.

6.5.4 Previous Values

PreviousValues represents the previous values that the parameter has accepted.

> **Note**
>
> They are only returned by the Web Intelligence RESTful Web Service SDK.

PreviousValues is structured as default values:

```
<previous>
    <value id="string" path="[[0|1|2,\ second_level,\ index],[0|1|2,\
third_level,\ index],[...]]">
    ...
```

The `GET .../parameters` request adheres to the following rules regarding the management of previous and default values:

- Previous values are returned inside `<answer>/<values>` when they exist.
- If there are no previous values but default values, then default values are returned inside `<answer>/<values>`.
- If there are no previous nor default values, then nothing is returned.

6.5.5 Incompatible Contexts

ContextIncompatibilities represents the contexts that are incompatible.

ContextIncompatibilities is structured as follows:

```
<incompatibility>
```

```
    <values>
        <value id="string">
        <value id="string">
    ...
```

Context values are grouped by pair.

Element	Type or Value	Description
`<value>`	*string*	The context value

The following table describes the `<value>` attribute:

Attribute	Type or Value	Description
`id`	*string*	The context value identifier

6.5.6 Answer Request Body Schemas

Request Body Schemas (XML)

`(PUT .../parameters)`

This schema contains the actual answers to pass to contexts or prompts, and the query of a list of values for the parameters that remain to be answered.

```
<parameters>
    <parameter>
        <id>
        <answer type="Text|Numeric|Date|Unknown" constrained="Boolean">
            <info>
            <values>
                <value path="[0|1|2,\ level,\ index]" id="integer">
```

```
<info>
    <lov>
        <query intervalId="integer" intervalSize="Integer|-1|Unlimited|Server"
refresh="Boolean">
            <sort order="Boolean"/>
            <search>
            <path>
                <value id="integer" type="String|Date|Numeric">
```

Parameter

Element	Type or Value	Description
`<id>`	*integer*	Parameter identifier
`<answer>`	N/A	The actual answer

Answer

Element	Description
`<info>`	Contains one of the following: • The possible values of the list of values. See its description in Lists of Values [page 136]. • The query used to retrieve a list of values for unanswered parameters. The query can specify how values will be returned, and/or refreshed if the list of values allows it. → Remember Not mandatory in the request bodies.
`<values>`	The list of actual values expressed as a sequence of `<value>` elements. Is ignored by the service in the request bodies.

Values

Element	Type or Value	Description

`<value>`	*string*	An actual value
		Attributes:
		• `id`: Defines the context identifier or the value index. Mandatory if the value contains an answer to a context or if the prompt is index-aware.
		• `path`: `[[0|1|2,\ level,\ index], [...]]`. Optional. In the case of hierarchical parameters, specifies the types, values, and indexes of the levels of the hierarchy, starting with the second level. The call needs this information to reach the answer value.
		The syntax `[0|1|2,\ level,\ index]` describes the data type, value, and index of a level. Indexes are mentioned if the hierarchy has indexed levels.
		  • `0` represents a string
		  • `1` represents a date
		  • `2` represents a number
		You can also use `String`, `Date`, or `Numeric` explicitely instead of numbers.
		See examples in Example - Responding to a Hierarchical Parameter [page 196] and Example - Refreshing a Document with a Hierarchical Parameter [page 633].

Query

Attribute	Type or Value	Description
`intervalId`	*integer*	The index of the interval that should be returned. If not specified,the first interval is returned. An error is returned if this index is out of range (depending of values count).
`intervalSize`	*integer*|`-1`| `Unlimited`|`Server`	Specifies the number of values in the interval to return. If not specified, 50 is used. Possible values are: • A strictly positive integer • `-1` or `Unlimited` indicates the whole list of values is returned • `Server` indicates that the Information Engine Service property *List of Values Batch size (entries)* of the CMC defines the number of values to return.
`refresh`	*Boolean*	Specifies whether the list of values is refreshed. Optional. An error is returned if the list of values does not allow refreshing.

Element	Type or Value	Description
`<sort>`	*Boolean*	Specifies whether the list of values must be returned sorted. Optional. **Attribute**: `order` (`Ascending`|`Descending`|`None`)
`<search>`	*string*	Optional. Defines a search pattern in the list of values as a string. The following wildcard characters may be used in the pattern string: `?` for zero or one character, and `*` for zero or n characters. For example, `M?Gregor` yields to the value `McGregor`, and `M*Gregor` to the values `McGregor` and `MacGregor`.
`<path>`	N/A	Optional. In the case of a hierarchical parameter, it specifies the intermediate answer value. It also specifies the node for which the list of values is requested. This path is expressed as a sequence of `<value>` elements. See examples in Example - Responding to a Hierarchical Parameter [page 196] and Example - Refreshing a Document with a Hierarchical Parameter [page 633].

Query Path

Element	Type or Value	Description
`<value>`	*string*	The value for a node of the path. **Attributes:** • `id`: defines the context identifier or the value index. Mandatory if the value contains an answer to a context or if the prompt is index-aware. • `type`: The value type (`String`|`Date`|`Numeric`). Default is `String`.

6.6 Change Source

Changing data sources of a query in a Web Intelligence document means replacing data objects from a data source with data objects from another data source. This replacement is computed according to a set of rules called strategies, which define how target data objects should match the data objects to be replaced.

Use Cases

The "change source" functionality addresses two main use cases:

- Linking a document to a universe that has been converted from UNV to UNX
- Linking a document uploaded to the CMS repository to a data source actually stored in the CMS repository

Supported Data Sources

The following table describes the source and target data sources supported by the "change source" functionality.

	Target UNV	Target UNX	Target BEx Query	Target SAP HANA Direct Access
Source UNV	✔	✔	✔	✔
Source UNX	✘	✔	✔	✔
Source BEx Query	✘	✔	✔	✔

	Target UNV	Target UNX	Target BEx Query	Target SAP HANA Direct Access
Source SAP HANA Direct Access	✗	✓	✓	✓

> **Note**
> - "UNX" includes relational, OLAP and BEx query-based UNX universes.
> - "SAP HANA Direct Access" includes relational and OLAP SAP HANA data sources.
> - The "change source" functionality does not support text files, Microsoft Excel spreadsheets, SAP HANA Online and Web Services as data sources.

"Change Source" Worflow

Before 4.1 SP4

1. GET: getting the suggested data object mappings for the selected data providers based on a given target data source.
   The default strategy applies.
2. POST: updating the document with the selected data objects of the target data source mapped to the data objects to be replaced.
   The object mapping used can be the one suggested by the first call or can have been edited.

Since 4.1 SP4 Patch 4 and 4.1 SP5 Patch 1

It is possible to apply the suggested mapping (POST call) without getting it (GET call). This means the mapping is considered to be valid.

In the POST call:

- If the request body is empty, then the object mapping used is the one found by the default strategies.
- If the request body contains an incomplete object mapping, then the default mapping is used to replace the missing objects.

Since 4.1 SP6

1. Getting the suggested data object mappings using either the default strategies (GET) or the given strategies (PUT).
   In the GET call, the default strategies are the ones used in previous releases. In the PUT call, you specify the order of the strategies, which apply one after the other for each data object only when the mapping status is "not found" and until a match is found.
2. POST: updating the document with the selected data objects of the target data source mapped to the data objects to be replaced.
   The object mapping used in the POST call can be the one suggested either by the GET call (default mapping) or by the PUT call (mapping found by the given stragtegies). A custom mapping can also be specified in the request body.
   In the POST call:
   - If the request body is empty, then the object mapping used is the default mapping.
   - If the request body contains an incomplete object mapping, then the default mapping is used to replace the missing objects.

- If the request body contains an empty or incomplete object mapping but with strategies, then the mapping used is overloaded by the one found with the strategies.

> **Note**
>
> - For custom mappings, you can only change the target ID of a given mapping, but must keep all the source IDs. If the target ID is null, then the data source object will be removed.
> - Before 4.2 SP3, the ID of a data provider affected by a "change source" was modified from N to N+1, for example from `DP0` to `DP1`. It is no longer the case since 4.2 SP3.

Mapping Request and Response Body Schemas [page 147]

Strategies [page 149]

Compatibility Rules [page 150]

6.6.1 Mapping Request and Response Body Schemas

Request Body (XML)

(POST method)

```
<mappings>
    <policy qualificationTolerance="Low|Normal|High" dataTypeTolerance="Low|
Normal|High">
        <strategies mappingSourceIds="string">
            <strategy name="SameId|SameName|SameTechnicalName|SamePath|CloseName|
Selection|Removal" targetId="string" />
    <content>
        <mapping>
            <source>
                <id>
            <target>
                <id>
    <parameters>
        <parameter>
```

Element	Description
`<policy>`	**Attributes**: • `qualificationTolerance` (`Low|Normal|High`) • `dataTypeTolerance` (`Low|Normal|High`) They specify the tolerance to be considered when matching the source and target datasource objects. They define the compatibility rules.
`<strategies>`	The list of strategies to apply. This element is optional. **Attribute**: `mappingSourceIds` to apply the strategies to the specified source data objects. The IDs are separated by a comma. If no attribute is specified, the strategy will apply to all source data objects (default strategy).

Element	Description
`<strategy>`	A strategy selected to compute the possible object mappings. **Attributes**: • `name`: strategy name • `targetId` required by the `Selection` strategy to specify the ID of the target data object to be selected
`<content>`	The custom object mapping. This element is optional.
`<parameters>`	The data source parameters if necessary (SAP HANA variables and BEx query variables). This element is optional. See Answer Request Body Schemas [page 142].

The following table describes the `qualificationTolerance` attribute values:

Attribute value	Description
`Low`	The source and target data objects must have the same qualification.
`Normal`	The source and target data objects can have close qualifications.
`High`	The source and target data objects can have different qualifications.

The following table describes the `dataTypeTolerance` attribute values:

Attribute value	Description
`Low`	The source and target data objects must have the same data type.
`Normal`	The source and target data objects can have close data types.
`High`	The source and target data objects can have different data types.

Example

`qualificationTolerance="Low" dataTypeTolerance="High"` means:

- The qualifications of the source and target data objects must be exactly the same.
- The data types of the source and target data objects can be different.

Response Body (XML)

(GET and POST methods)

```
<mappings>
    <content>
        <mapping status="Ok|Ambiguous|Not found">
```

```
            <source>
                <id>
            <target>
                <id>
```

Element	Type or Value	Description
`<mapping>`	N/A	**Attribute**: `status` • `Ok` if the mapping fully matches • `Ambiguous` in case of uncertain mapping • `Not found` if there is no possible match **Note** The `Not found` status removes the corresponding ID from the document when doing the data source change.
source `<id>`	*string*	The identifier of the source data object
target `<id>`	*string*	The identifier of the target data object

6.6.2 Strategies

A strategy is a rule that defines how an object of the current data source can match an object of the target data source. A strategy allows you to find the data object mapping.

The following table describes the strategies that can apply:

Strategy	Description
"SameID"	Searches a valid object that has the same ID
"SameTechnicalName"	Searches a valid object that has the same technical name **Note** This strategy applies to BEx Queries and BEx Query-based UNX universes only.
"SamePath"	Searches a valid object that has the same path and type. The path includes the object name and its parent folders.
"CloseName"	Searches a valid object that has the closest name based on the Levenshtein distance, with the distance lower than a fixed threshold. If several objects have the same distance, then the selected object is the one whose ID comes first in alphabetical order.
"Removal"	Removes the object from the data provider if the previous strategies did not find any target object
"Selection"	Selects a specific target object. Requires the attribute "targetId" attribute containing the ID of the target data object to be selected
"SameName"	Maps a valid object whose Levenshtein distance is null, that is, finds an object with the same name

The following default strategies apply in the following order to find what is called the default mapping:

1. "SameID"
2. "SameTechnicalName"
3. "SamePath"
4. "CloseName"

Related Information

Getting the Possible Object Mappings Using the Default Strategies [page 573]
Getting the Possible Object Mappings Using Selected Strategies [page 576]

6.6.3 Compatibility Rules

When running the "change source", the following compatibility rules apply to avoid an inconsistent mapping:

- Compatibility of the source and target object types (dimension, measure, attribute, hierarchy, and so on)
- Compatibility of the source and target object data type (member, string, numeric, date, and so on)
- Compatibility of the source and target object usage, if the object can be used as a result object, a filter object, or a sort object

These compatibility rules are defined through the `qualificationTolerance` and `dataTypeTolerance` `<policy>` attributes. They specify the tolerance to be considered when matching the source and target objects.

The following table shows the possible object matches according to the `qualificationTolerance` attribute value applied to the object type.

	Target Dimension	Target Hierarchy	Target Level	Target Dimension Attribute	Target Measure Attribute	Target Measure	Target Others
Source Dimension	Low	Medium	Medium	Medium	Medium	High	High
Source Hierarchy	High	Low	High	High	High	High	High
Source Level	High	High	Low	High	High	High	High
Source Dimension Attribute	Medium	High	High	Low	High	High	High

	Target Dimension	Target Hierarchy	Target Level	Target Dimension Attribute	Target Measure Attribute	Target Measure	Target Others
Source Measure Attribute	Medium	High	High	High	Low	High	High
Source Measure	High	High	High	High	High	Low	High
Source Others	High	High	High	High	High	High	Low if the same, else High

> **Note**
>
> Other object types are Predefined filters, Named sets, Calculated members, Prompt parameters, and universe lists of values.

The following table shows the possible object matches according to the `dataTypeTolerance` attribute value applied to the data type.

	Target N/A	Target Member	Target Numeric	Target String	Target Time/Date	Target Others
Source N/A	Low	High	High	High	High	High
Source Member	High	Low	Medium	Medium	Medium	Medium
Source Numeric	High	Medium	Low	High	High	High
Source String	High	Medium	High	Low	High	High
Source Time/Date	High	Medium	High	High	Low if the same, else Medium	High
Source Others	High	Medium	High	High	High	Low if the same, else High

> **Note**
>
> - N/A refers to the objects of type Filter and NamedSet, which do not have any data type.
> - Time/Date data types are Date, DateTime, Time, and CalendarDate.
> - Other data types are Blob, Boolean, DataUnknown, Currency, Null, Percent, Scientific, and LongText.

6.7 Schedules

Body Schema (XML)

(GET .../documents/<documentID>/schedules/<scheduleID>)

(POST .../documents/<documentID>/schedules)

```
<schedule>
    <id>
    <name>
    <format type="webi|pdf|xls|csv|txt|csvArchive|htmlArchive"/>
    <deliveryRules>
    <status id="0|1|3|8|9|13|14">
    <updated>
    <destination>
    <error>
    recurrence_expression
    <serverGroup id="integer" required="Boolean"/>
    <parameters>
```

Schedules

Element	Type or Value	Description
`<id>`	*integer*	The schedule identifier **Note** This element only appears in the response of the `GET` schedule details call.
`<name>`	*string*	The schedule name
`<format>`	N/A	The document format. **Attribute**: `type` (`webi|pdf|xls|csv|txt|csvArchive|htmlArchive`). By default, the format is `webi`. See the next section for options specific to the `csv` format.

Element	Type or Value	Description

`<deliveryRules>`
`  <deliveryRule>`
`  <id>`
`  <status>`
	N/A	The delivery rules of the schedule. Add this tag only if you want to send the document if it has been successfully refreshed, or contains data, or both.
		• `<id>` is the name of the delivery rule, that is `ContainsData` or `FullyRefreshed`.
		• `<status>` corresponds to the status of the instance when the delivery rule is not met. Possible values are `Warning` or `Failed`.

`<status>`	*string*	The schedule status. **Attribute**: `id` Each `id` can help you to test the schedule independently of your language locale. • `<status id="0">Running</status>` • `<status id="1">Completed</status>` • `<status id="3">Failed</status>` • `<status id="8">Paused</status>` • `<status id="9">Pending</status>` • `<status id="13">Expired</status>` • `<status id="14">Warning</status>` **Note** This element only appears in the response of the `GET` schedule details call.
`<updated>`	*string*	The timestamp of the last schedule

Element	Type or Value	Description
`<destination keepInstanceInHistory="Boolean">` `<useSpecificName fileExtension="Boolean">`	N/A	The document destination, which can be one of the following: • The BI launch pad inbox • An email address (SMTP) • An email address over SSL (SMTPS) • A local file • A file via FTP or secured FTP (SFTP) Is optional. If no destination is specified, then the default destination is the one defined by the administrator on the BI platform via the CMC (*Servers* > *Web Intelligence Services* > *Adaptive Job Server* > *Destination*). See the examples to learn how to set up the destination in the request body. **→ Remember** To schedule to an SMTPS server, make sure you have configured the BI platform properly. See KBA n° 2263613 "How to set up SMTP over SSL". **Attribute**: `keepInstanceInHistory` (Boolean) keeps the instance of the schedule document in history if it is `true` (default) • `<useSpecificName>` is the specific name of the scheduled document at destination. A name is assigned automatically if the element is not specified. • `fileExtension` specifies a file extension. Default is `true`.

`<error>`
`<error_code>`
`<message>`
	N/A	Error information added in case of failure (`<status id="3">Failed</status>`).
		Note
		This element can only appear in the response of the `GET` schedule details call.

recurrence_expression	N/A	The scheduling recurrence expression. See the next section to learn how to set it up.
`<parameters>`	N/A	The parameter used as prompt. See the Refreshing Documents [page 619] for information on the `<parameters>` details. See Example - Inbox Destination for a Daily Schedule with a Prompt [page 674].

Element	Type or Value	Description
`<serverGroup id="integer" required="Boolean"/>`	N/A	The server group used to run the schedule **Attributes**: • `id` is the server group ID as defined in the CMC (*integer*). • `required`: *Boolean*. If `false`, the schedule is run preferably on the server that belongs to the specified server group. If the specified server is not available, then the schedule is run on the next available server. If `true`, only the specified servers found within the selected server group is used. If all of the servers in the server group are unavailable, then the schedule is not processed.

CSV Document Format

If the format type is CSV, the following options are supported:

```
<format type="csv">
    <properties>
        <property key="textQualifier">'</property> <!-- either ' or " -->
        <property key="columnDelimiter">,</property> <!-- either , or ; or tab
-->
        <property key="charset">UTF-8</property> <!-- a valid server charset -->
        <property key="onePerDataProvider">false</property>
    </properties>
</format>
```

Recurrence Expressions

The following table describes the possible recurrence expressions.

Recurrence Expression	Schedule Time
No recurrence expression	Now

Recurrence Expression	Schedule Time

`<once retriesAllowed="2" retryIntervalInSeconds="60">`
`  <startdate>`
`  <enddate>`
	Once, run at `<startdate>`.
	• `retriesAllowed`: the number of retries (*integer*)
	• `retryIntervalInSeconds`: the time between two retries (*integer*)
	• `<startdate>`: the date and time to run the schedule (*dateTime*)
	• `<enddate>`: the date and time to stop the schedule (*dateTime*)

`<daily retriesAllowed="2" retryIntervalInSeconds= "60">`
`  <startdate>`
`  <enddate>`
`  <dayinterval>`
	Daily.
	`<dayinterval>` is the day interval to repeat the schedule (*integer*).

`<hourly retriesAllowed="2" retryIntervalInSeconds="60">`
`  <startdate>`
`  <enddate>`
`  <hour>`
`  <minute>`
	Hourly.
	`<hour>` (*integer*, [0, 24]) and `<minute>` (*integer*, [0,59]) define the time interval to repeat the schedule.

`<weekly retriesAllowed="2" retryIntervalInSeconds="60">`
`  <startdate>`
`  <enddate>`
`  <monday/>`
`  <tuesday/>`
`  <wednesday/>`
`  <thursday/>`
`  <friday/>`
`  <saturday/>`
`  <sunday/>`
	Weekly. The schedule is run only the mentioned days.
	`<monday/>`, `<tuesday/>`, `<wednesday/>`, `<thursday/>`, `<friday/>`, `<saturday/>`, and `<sunday/>` are all optional. They must be mentioned in the specified order.

`<monthly retriesAllowed="2"retryIntervalInSecon ds= "60">`
`  <startdate>`
`  <enddate>`
`  <month>`
	Monthly.
	`<month>` defines the month interval to repeat the schedule (*integer*, [1, 12]).

`<nthDayOfMonth retriesAllowed="2" retryIntervalInSeconds="60">`
`  <startdate>`
`  <enddate>`
`  <day>`
	Only the specified day of each month.
	`<day>` defines the day of the month as a number between 1 and 31.

`<firstMondayOfMonth retriesAllowed="2" retryIntervalInSeconds="60">`
`  <startdate>`
`  <enddate>`
	The first Monday of each month.

Recurrence Expression	Schedule Time

`<lastDayOfMonth retriesAllowed="2" retryIntervalInSeconds="60">`
`<startdate>`
`<enddate>`
	The last day of each month.

`<xDayOfNthWeekOfTheMonth retriesAllowed="2" retryIntervalInSeconds="60">`
`<startdate>`
`<enddate>`
`<day>`
`<week>`
	The specified day of the specified week of each month.
	`<day>` defines the day of the month as a number between 1 and 31.
	`<week>` defines the week of the month as a number between 1 and 4.

`<calendar retriesAllowed="2" retryIntervalInSeconds="60">`
`<startdate>`
`<enddate>`
`<id>`
	The schedule runs according to the recurrence defined in the calendar that the end-user has created on the CMC.
	`<id>` is the calendar object identifier in the CMS.

Related Information

Example - File System Destination for a Now Schedule [page 671]
Example - FTP Destination for a Once Schedule [page 672]
Example - SFTP Destination for an Hourly Schedule [page 672]
Example - Mail Destination for a Hourly Schedule [page 673]
Example - Inbox Destination for a Daily Schedule with a Prompt [page 674]
Example - Inbox Destination for a Monthly Schedule to Specific Users [page 675]
Example - Schedule for a Server Group [page 676]
Getting the Details of a Schedule [page 669]

6.8 Universes

> → Remember
>
> The RESTful Web Service SDKs support relational universes created with the information design tool (UNX) and with the universe design tool (UNV).

Response Body Schema (XML)

(GET .../universes/<universeID>)

```
<universe>
    <id>
    <cuid>
    <name>
    <description>
    <type>
    <folderId>
    <path>
    <maxRowsRetrieved>
    <maxRetrievalTime>
    <connected>
    <outline aggregated="Boolean">
        <businessViewName>
        <item>
        <folder>
            <id>
            <name>
            <description>
            <customProperty name="string">
            <item>
            <folder>
```

Universe

Element	Type or Value	Description
<id>	*integer*	The universe identifier
<cuid>	*string*	The unique CUID of the universe in the CMS repository
<name>	*string*	The universe name in the requested locale
<description>	*string*	The universe description
<type>	unv|unx	The universe type
<folderId>	*integer*	The ID of the folder that contains the universe in the CMS repository
<path>	*string*	The universe path in the CMS repository (returned if the GET call is made with the Web Intelligence RESTful Web Service SDK)
<maxRowsRetrieved>	*integer*	An option that defines the restrictions from the data source that can apply to queries
<maxRetrievalTime>	*integer*	An option that defines the restrictions from the data source that can apply to queries
<connected>	*Boolean*	`true` if the connection with the RDBMS is set (returned if the GET call is made with the Web Intelligence RESTful Web Service SDK)

Element	Type or Value	Description
`<outline>`	N/A	The business layer details. **Attribute:** `aggregated` (*Boolean*) • If `aggregated` is `true`, the outline is aggregated. • If `aggregated` is `false`, the outline is the master view if it is granted, or the default view if the master view is denied.

Outline

Element	Type	Description
`<businessViewName>`	*string*	The name of the default view if the master view is denied
`<item>`	N/A	An object of the universe
`<folder>`	N/A	A UNX universe folder or UNV universe class

Item

```
<item type="string" dataType="string" hasLov="Boolean" forResult="Boolean"
forFilter="Boolean" forSort="Boolean">
    <id>
    <name>
    <description>
    <customProperty name="string">
    <item>
    <path>
    <aggregationFunction>
```

Element	Type	Description
`<id>`	*string*	The object identifier
`<name>`	*string*	The object name
`<description>`	*string*	The object description
`<customProperty>`	*string*	Optional custom property for UNX universes only **Attribute:** `name` (*string*)
`<item>`	N/A	A child object
`<path>`	*string*	The object full path, which adheres to some particular rules. See Object Full Paths [page 161].

Element	Type	Description
`<aggregationFunction>`	`Average|Sum|Count|First|Last|Max|Min|Delegated|none`	The aggregation function used for measures only. Default is `none`.

> **Note**
>
> These attributes cannot be used with folders and predefined filters.

Attribute	Type or Value	Description
`type`	*string*	The object type in the universe (dimension, measure, attribute, and so on)
`dataType`	`String|Numeric|DateTime`	The object data type
`hasLov`	*Boolean*	Specifies whether the object has a list of values (for dimensions, measures, attributes, levels and hierarchies)
`forResult`	*Boolean*	Specifies whether the object can be used as a result object (default is `true`)
`forFilter`	*Boolean*	Specifies whether the object can be used as filter (default is `true`)
`forSort`	*Boolean*	Specifies whether the object can be used to sort data (default is `true`)

Folder

Element	Type or Value	Description
`<id>`	*string*	The folder identifier (for UNX universes only)
`<name>`	*string*	The folder name
`<description>`	*string*	The folder description
`<customProperty>`	*string*	Optional custom property for UNX universes only **Attribute:** `name` (*string*)
`<item>`	N/A	An object of the folder
`<folder>`	N/A	A subfolder

Related Information

Getting the Details of a Universe (Semantic Layer) [page 168]
Getting the Details of a Universe (Web Intelligence) [page 696]

6.9 Object Full Paths

In XML request or response bodies such as a query specification or the description of a UNV or UNX universe, the objects of the universe are identified by the following attributes:

- `id`
  This is the object identifier as defined in the CMS repository.
- `path`
  This is the full path of the object. This attribute is optional and is mainly used by client tools to implement universe change source. If the universe data source must be replaced with another one containing different object IDs, then only the object paths can be used to map the objects in the query and the objects in the new universe data source.

You build the full paths by following the rules below:

- Each node in the path is made of the object name and type, separated by `|`: "`<name>|<type>`".
- The escape character of `|` and ~ is ~.
- The nodes of the path are concatenated with `\`: "`Age Group|folder\Age Max|dimension`".
- The escape character of `\` and § is §.

The following table shows the possible types of objects manipulated in each of the provided SDKs:

SDK	Object Type
BI Semantic Layer RESTful Web Service SDK	• attribute • dimension • filter • folder • measure
Web Intelligence RESTful Web Service SDK	• analysisDimension • attribute • aggregationAttribute • calculatedMember • dimension • filter • folder • hierarchy • level • measure • namedSet

Example

The following examples illustrate the naming rules:

Path	Object	String
Root folder	"CustomerName" dimension	"`CustomerName|dimension`"
"Customer" folder	"Name" dimension	"`Customer|folder\Name|dimension`"
"Customer" dimension in "Contact" folder	"Name" attribute	"`Contact|folder\Customer|dimension\Name|attribute`"
"Customer\Large" dimension in "Country|US" folder	"First~Name" attribute	"`Country~|US|folder\Customer§\Large|dimension\First~~Name|attribute`"

6.10 User Rights

User Rights Applied to UNV Universes

User rights that applied to UNV universes fall into three categories:

- Security rights
  Security rights can be set at universe design tool, universe and relational connection level.
- Universe overloads
  The rights defined on universes are Connections, Controls, SQL, Objects, Rows, and Table Mapping.
- Object access level
  The object access level defines a restriction level for users (Private, Public, Confidential...) A user who does not have the required access level cannot view the objects beyond his scope.

These user rights are applied before an application user makes use of the REST APIs and therefore are transparent to the REST API user. For example, objects denied by the access level are not returned in the universe outline, when getting it through `GET .../universes/<universeID>`. Connections, Objects, Rows, and Table Mapping rights are applied when getting the query results through the OData service or when getting the list of values through `GET .../queries/<queryID>/parameters`.

Only a few security rights can have an impact on the use of the REST APIs. The following table presents them.

Right	Description
"View objects" at relational connection level	If this right is disabled, the user cannot see the connection and the query cannot be run.
"View objects" at universe level	If this right is disabled, the universe is not displayed in the universe list and any call with the universe ID returns an error.
"Data access"	This custom security right is defined at universe level or relational connection level. If this right is disabled, any call to the OData service returns an error.

User Rights Applied to UNX Universes

The same user rights apply to UNX universes. Universe overloads are managed through business and data security profiles. Business security profiles manage rights on metadata at the business layer level. Data

security profiles manage rights at the data foundation level. These profiles are attached to users or groups of users. The same security rights can have an impact on the use of the REST APIs.

6.11 Getting the Web Intelligence Security Rights for the Current User

Usage

This URL returns the list of allowed security rights of the current session.

Request

`GET /session/rights`

Response

Response type: `application/xml` or `application/json`.

The available rights are:

- `add_objects_to_folder`
- `allow_personal_data_providers`
- `app_allow_autosave`
- `app_create_edit_input_controls`
- `app_edit_document_formatting`
- `app_edit_my_preferences`
- `app_enable_context_menus`
- `app_hide_show_toolbox`
- `app_lp_document_filters`
- `app_lp_document_summary`
- `app_publish_document_content`
- `app_re_create_edit_alerter`
- `app_re_create_edit_break`
- `app_re_create_edit_predefined_calculation`
- `app_re_create_edit_report_filters`
- `app_re_create_edit_sort`

- app_re_insert_element
- app_view_sql
- change_TDC_format
- create_documents
- delete_objects
- disable_automatic_refresh_on_open
- disable_export_to_BIOD
- disable_import_from_BIOD
- edit_documents
- edit_objects
- edit_query_sql
- enable_shared_objects
- insert_shared_objects
- merge_for_synchro
- publish_documents_real
- rc_enable_use
- rc_export
- rc_import
- rc_local_save
- rc_print
- rc_remove_document_security
- rc_save_for_all_users
- rc_send_by_mail
- read_corporate_documents
- use_TDC
- use_formula_language
- use_interactive_viewing

Example: XML Format

Response

GET /session/rights

```
<?xml version="1.0" encoding="UTF-8"?>
<rights>
    <right>
        <id>use_interactive_viewing</id>
    </right>
    <right>
        <id>edit_documents</id>
    </right>
    <right>
        <id>publish_documents_real</id>
    </right>
    ...
</rights>
```

Example: JSON format

Response

`GET /session/rights`

```
{
    "rights": {
        "right": [{
            "id": "use_interactive_viewing"
        },
        {
            "id": "edit_documents"
        },
        ...
        }]
    }
}
```

7 BI Semantic Layer REST API Reference

Reference of the BI Semantic Layer REST APIs.

Each API reference provides the following information:

- What the API does
- URL of the HTTP request
- Description of the request query parameters
- Request body if needed (XML or JSON example used)
- Response type, which determines the value of the `accept` header attribute
- Response body (not described if the response is a success or error message)
- Response examples

Managing Universes with the BI Semantic Layer RESTful Web Service SDK [page 166]

Managing Queries [page 179]

Managing Parameters [page 184]

Managing Query Results [page 205]

7.1 Managing Universes with the BI Semantic Layer RESTful Web Service SDK

The BI Semantic Layer RESTful Web Service SDK provides a series of APIs to perform on UNX and UNV universes.

Getting the List of the Universes (Semantic Layer) [page 166]

Getting the Details of a Universe (Semantic Layer) [page 168]

Getting the Business Views of a Universe [page 171]

Getting the Query Capabilities of a Universe (Semantic Layer) [page 174]

Getting the Link Groups of a Universe [page 176]

Getting the Object Parameters of a Universe [page 178]

7.1.1 Getting the List of the Universes (Semantic Layer)

Usage

Gets the list of UNX and UNV universes stored in the CMS repository.

The universe list can be displayed on pages.

Request

`GET /universes?offset=<offset>&limit=<limit>`

Where:

- `<offset>` indicates the position in the list, from which universes are returned. It must be greater than or equal to 0. The default value is 0. This parameter is optional.
- `<limit>` indicates the number of universes that you can list on one page. The range is [1, 50]. The default value is 10. This parameter is optional.

Response

Response type: `application/xml`

Response body: the list of the universes identified by the following elements:

- `<id>`
- `<cuid>`
- `<name>` in the requested locale
- `<type>` (`unv` or `unx`)
- `<folderId>`, which is the ID of the folder that contains the universe in the CMS repository

Example

`GET /universes?offset=0&limit=50`

```
<universes>
    <universe>
        <id>6773</id>
        <cuid>AXyRzvmRrJxLqUm6_Jbf7lE</cuid>
        <name>efashion.unv</name>
        <type>unv</type>
        <folderId>6771</folderId>
    </universe>
    <universe>
        <id>5808</id>
        <cuid>AUW2qRdU0IdPkyhlpZWrxvo</cuid>
        <name>Warehouse.unx</name>
        <type>unx</type>
        <folderId>5807</folderId>
    </universe>
    ...
</universes>
```

7.1.2 Getting the Details of a Universe (Semantic Layer)

Usage

Gets the details of a UNX or UNV universe referenced by its ID.

Request

`GET /universes/<universeID>?aggregated=<aggregated>`

Where:

- `<aggregated>` is an optional, Boolean parameter that indicates if the outline must be aggregated. Default value is `false`.

Influence of the aggregated Parameter

The UNX universe details returned depend on the `<aggregated>` parameter value:

- If `false`, the call returns the master view if granted or the default view if the master view is denied. The default view name is returned in the outline using the `<businessViewName>` element. This behavior is the one implemented in the SDK versions prior to 4.1 SP5.
- If `true`, the call returns an aggregated outline containing all folders and objects granted to the user. This outline merges all granted objects from the granted business view and contains all objects properties such as ID, path, and name.

If the master view is granted, then the outline returned with the calls `.../<universeID>?aggregated=false` and `.../<universeID>?aggregated=true` are the same, except the value of `aggregated` outline attribute.

Even if UNV universes do not support the concept of view introduced with UNX universes, for API consistency, the BI Semantic Layer RESTful Web Service can also behave as if a UNV universe had only one master view containing all the universe content. When returning this view, the `<aggregated>` parameter has no impact on the response and the value of the `aggregated` outline attribute is `false` in the response.

Response

Response type: `application/xml` or `application/json`

Response body: the details of the universe. See Universes [page 157].

Example: The Master View is Granted

`GET /universes/5808?aggregated=false`

or

`GET /universes/5808`

Response:

```
<universe>
    <id>5808</id>
    <cuid>AUW2qRdU0IdPkyhlpZWrxvo</cuid>
    <name>Warehouse.unx</name>
    <type>unx</type>
    <folderId>5807</folderId>
    <maxRowsRetrieved>5000</maxRowsRetrieved>
    <maxRetrievalTime>600</maxRetrievalTime>
    <outline aggregated="false">
        <folder>
            <id>_IBo8OLIhEeCk0Ylv-tlF2Q</id>
            <name>Customer</name>
            <item dataType="Numeric" type="Dimension" hasLov="false">
                <id>_IBo8ObIhEeCk0Ylv-tlF2Q</id>
                <name>Customer Id</name>
                <path>Customer|folder\Customer Id|dimension</path>
            </item>
            <item dataType="String" type="Dimension" hasLov="false">
                <id>_IBo8OrIhEeCk0Ylv-tlF2Q</id>
                <name>Customer</name>
                <path>Customer|folder\Customer|dimension</path>
           </item>
            <folder>
                <id>_IBo8LbIhEeCk0Ylv-tlF2Q</id>
                <name>Geography</name>
                <item dataType="Numeric" type="Dimension" hasLov="false">
                    <id>_IBo8MrIhEeCk0Ylv-tlF2Q</id>
                    <name>Continent Id</name>
                    <path>Customer|folder\Geography|folder\Continent Id|
dimension</path>
                </item>
                <item dataType="String" type="Dimension" hasLov="false">
                    <id>_IBo8M7IhEeCk0Ylv-tlF2Q</id>
                    <name>Continent</name>
                    <path>Customer|folder\Geography|folder\Continent|dimension</
path>
                </item>
                ...
            </folder>
        </folder>
           ...
        <folder>
            <id>_IB8eE7IhEeCk0Ylv-tlF2Q</id>
            <name>Inventory</name>
            <item dataType="Numeric" type="Measure" hasLov="false">
                <id>_IB8eFrIhEeCk0Ylv-tlF2Q</id>
                <name>Stock Level</name>
                <path>Inventory|folder\Stock Level|measure</path>
            </item>
            <item type="Filter">
                <id>_V5_GkLR_EeCZotjuycya1A</id>
                <name>Stock Below Minimum</name>
                <path>Inventory|folder\Stock Below Minimum|filter</path>
            </item>
        </folder>
        <folder>
```

```
            <id>_H4I54LaXEeCH2d6gof1MOA</id>
            <name>Quarterly KPIs</name>
            <item dataType="Numeric" type="Measure" hasLov="false">
                <id>_oGzSYLaaEeCH2d6gof1MOA</id>
                <name>Reference Quarter</name>
                <path>Quarterly KPIs|folder\Reference Quarter|measure</path>
                <aggregationFunction>Sum</aggregationFunction>
            </item>
            <item dataType="Numeric" type="Measure" hasLov="false">
                <id>_vsi7wLaaEeCH2d6gof1MOA</id>
                <name>Same Quarter Last Year</name>
                <path>Quarterly KPIs|folder\Same Quarter Last Year|measure</path>
                <aggregationFunction>Sum</aggregationFunction>
            </item>
            <item dataType="Numeric" type="Measure" hasLov="false">
                <id>_68SKoLaaEeCH2d6gof1MOA</id>
                <name>YTY Change</name>
                <description>Year-to-Year Change</description>
                <path>Quarterly KPIs|folder\YTY Change|measure</path>
                <aggregationFunction>Sum</aggregationFunction>
            </item>
            ...
        </folder>
    </outline>
</universe>
```

Example: The Master View is Denied

```
GET /universes/5808?aggregated=false
```

or

```
GET /universes/5808
```

Response:

```
<universe>
    <id>5808</id>
    <cuid>AUW2qRdU0IdPkyhlpZWrxvo</cuid>
    <name>Warehouse.unx</name>
    <type>unx</type>
    <folderId>5807</folderId>
    <maxRowsRetrieved>5000</maxRowsRetrieved>
    <maxRetrievalTime>600</maxRetrievalTime>
    <outline aggregated="false">
        <businessViewName>MyView</businessViewName>
        <folder>
            ...
    </outline>
</universe>
```

Example: The Call Requests an Aggregated Outline

```
GET /universes/5808?aggregated=true
```

Response:

```
<universe>
    <id>5808</id>
    <cuid>AUW2qRdU0IdPkyhlpZWrxvo</cuid>
    <name>Warehouse.unx</name>
    <type>unx</type>
    <folderId>5807</folderId>
    <maxRowsRetrieved>5000</maxRowsRetrieved>
    <maxRetrievalTime>600</maxRetrievalTime>
    <outline aggregated="true">
        <folder>
            ...
    </outline>
</universe>
```

Related Information

Getting the List of the Universes (Semantic Layer) [page 166]

7.1.3 Getting the Business Views of a Universe

Usage

Universe	Response
UNV	The API returns a view considered as the master view of a UNV universe.
UNX	The API returns the business views of a UNX universe. The method returns only the views that are granted to the user that has opened the session. The views, including the master view, which are denied because of security settings, are not returned. **Note** For each business view, the user can navigate through tree nodes to get the objects that can be displayed in the outline. Each tree node refers either to a folder or to an object of the business layer by its identifier.

Request

`GET /universes/<universeID>/businessviews`

Request type : `application/xml`

Response

Response type : `application/xml`

Response body:

Universe	Response Body
UNV	A view considered as the master view (`masterView="true"`) that contains all classes and objects of the universe with: • `<description>`. If the description is not available, the `<description>` element does not display. • `<folderRef>` refers to a class of the view. • `<itemRef>` refers to an object of the business layer. There is only one view.
UNX	The master view (`masterView="true"`) and custom views for a user (`masterView="false"`), identified with: • `<id>` • `<name>` • `<description>`. If the description is not available, the `<description>` element does not display. • `<folderRef>` refers to a folder of the business view. • `<itemRef>` refers to an object of the business layer. There is only one master view. There can be several custom views.

Example: Getting the View of a UNV Universe

`GET /universes/11572/businessviews`

```
<businessViews xmlns="http://www.sap.com/rws/sl/universe">
    <businessView masterView="true">
        <folderRef name="Resort">
            <itemRef id="DO39"/>
            <itemRef id="DO2"/>
            <itemRef id="DO4"/>
            <itemRef id="DO5"/>
            <itemRef id="DFc"/>
            <itemRef id="DF1"/>
            <itemRef id="DF2"/>
        </folderRef>
        <folderRef name="Sales">
            <itemRef id="DO2f"/>
            <itemRef id="DO31"/>
            <itemRef id="DO32"/>
            <itemRef id="DO33"/>
            <itemRef id="DO17"/>
        </folderRef>
        <folderRef name="Customer">
            <itemRef id="DO11"/>
            <itemRef id="DO12"/>
            <itemRef id="DO13"/>
            <itemRef id="DOd">
                <itemRef id="DO9"/>
                <itemRef id="DOe"/>
```

```
                <itemRef id="DO8"/>
            </itemRef>
            <itemRef id="DO19"/>
            <itemRef id="DF4"/>
            <itemRef id="DFa"/>
            <itemRef id="DFb"/>
            <folderRef name="Sponsor">
                <itemRef id="DO2b">
                    <itemRef id="DO26"/>
                    <itemRef id="DO2c"/>
                </itemRef>
            </folderRef>
        </folderRef>
        <folderRef name="Reservations">
            <itemRef id="DO1e"/>
            <itemRef id="DO34"/>
            <itemRef id="DO35"/>
            <itemRef id="DO36"/>
        </folderRef>
        <folderRef name="Measures">
            <itemRef id="DO3a"/>
            <itemRef id="DO7"/>
            <itemRef id="DO14"/>
            <itemRef id="DO25"/>
        </folderRef>
    </businessView>
</businessViews>
```


Example: Getting the Business Views of a UNX Universe

`GET /universes/5808/businessviews`


```
<businessViews xmlns="http://www.sap.com/rws/sl/universe">
    <businessView masterView="true">
        <id>PER__6FCF0NdGEeKfwPJJ7KJ5ow</id>
        <name>UniverseName</name>
        <folderRef id="_6HHwgtdGEeKfwPJJ7KJ5ow">
            <itemRef id="_6HHwg9dGEeKfwPJJ7KJ5ow"/>
            <itemRef id="_6HRhgNdGEeKfwPJJ7KJ5ow"/>
            <itemRef id="_6HRhgddGEeKfwPJJ7KJ5ow"/>
            <itemRef id="_6opwQNdGEeKfwPJJ7KJ5ow">
            <itemRef id="_rhz58Oo_EeKgAukdblEjyA"/>
            </itemRef>
        </folderRef>
        <folderRef id="_6HuNcddGEeKfwPJJ7KJ5ow">
            <itemRef id="_6H3XYNdGEeKfwPJJ7KJ5ow"/>
            <itemRef id="_6H3XYddGEeKfwPJJ7KJ5ow"/>
            <itemRef id="_6IBIYNdGEeKfwPJJ7KJ5ow"/>
            <itemRef id="_6IBIYddGEeKfwPJJ7KJ5ow"/>
        </folderRef>
    </businessView>
    <businessView masterView="false">
        <id>_ouYoYzJyEeOx7dXQSVnuKw</id>
        <name>ViewName</name>
        <description>View description</description>
        <folderRef id="_6HHwgtdGEeKfwPJJ7KJ5ow">
            <itemRef id="_6HHwg9dGEeKfwPJJ7KJ5ow"/>
            <itemRef id="_6HRhgNdGEeKfwPJJ7KJ5ow"/>
        </folderRef>
        <folderRef id="_6HuNcddGEeKfwPJJ7KJ5ow">
            <itemRef id="_6H3XYNdGEeKfwPJJ7KJ5ow"/>
            <itemRef id="_6H3XYddGEeKfwPJJ7KJ5ow"/>
        </folderRef>
```

```
    </businessView>
</businessViews>
```

Related Information

Getting the List of the Universes (Semantic Layer) [page 166]

7.1.4 Getting the Query Capabilities of a Universe (Semantic Layer)

Usage

Gets the query capabilities of a UNX or UNV universe, depending on the user rights.

These capabilities list the features supported by the data source when building the query from the query panel.

Request

`GET /universes/<universeID>/capabilities`

Request type: `application/xml`

Response

Response type: `application/xml`

Response body:

Capabilities	Feature	Description
General query capabilities	`<combinedQueriesSupported>`	`true` if the data source supports the combined queries.
	`<maxValuesForInList>`	The maximum number of items that can be set in a filter based on INLIST or NOT_INLIST operators.
Data processing capabilities	`<removeDuplicateRowsAvailable>`	
	`<samplingLevel>`	

Capabilities	Feature	Description
	`<maxRowRetrievedAvailable>`	
	`<maxRetrievalTimeAvailable>`	
Filter capabilities	`<supportedComparisonOperators>`	The list of comparison operators supported for filters based on dimensions, attributes or measures.
	`<supportedLogicalOperators>`	The list of logical operators supported to combine filters.
	`<supportedObjects>`	The list of object qualifications (dimensions, attributes, and measures) supported to build queries.
	`<supportedRankings>`	The list of supported ranking operators (top, bottom, topPercent, or bottomPercent)
	`<resultObjectInFilterSupported>`	`true` if an object used as a result object can be used in a filter.
	`<objectComparisonSupported>`	`true` if filter operands based on another object are supported.
	`<constantComparisonSupported>`	`true` if filter operands based on constant values are supported.
	`<subQueriesSupported>`	`true` if subquery filters are supported.
Result object capabilities	`<useAttributeSeparatelyAvailable>`	
	`<sortObjectsAvailable>`	`true` if query sorts from objects are supported.
	`<sortNonResultObjectsAvailable>`	`true` if query sorts from objects that are not used as result objects are supported.

Example

`GET /universes/5808/capabilities`

```
<capabilities xmlns="http://www.sap.com/rws/sl/universe">
    <GeneralCapabilities>
        <combinedQueriesSupported>true</combinedQueriesSupported>
        <maxValuesForInList>-1</maxValuesForInList>
    </GeneralCapabilities>
    <DataProcessingCapabilities>
        <removeDuplicateRowsAvailable>true</removeDuplicateRowsAvailable>
        <samplingLevel>None</samplingLevel>
        <maxRowRetrievedAvailable>true</maxRowRetrievedAvailable>
        <maxRetrievalTimeAvailable>true</maxRetrievalTimeAvailable>
    </DataProcessingCapabilities>
    <filterCapabilities>
        <supportedComparisonOperators>
            <value>EqualTo</value>
            <value>NotEqualTo</value>
            <value>Between</value>
            <value>NotBetween</value>
```

```
            ...
        </supportedComparisonOperators>
        <supportedRankings>
            <value>top</value>
            <value>bottom</value>
            <value>topPercent</value>
            <value>bottomPercent</value>
        </supportedRankings>
        <supportedLogicalOperators>
            <value>or</value>
            <value>and</value>
        </supportedLogicalOperators>
        <supportedObjects>
            <value>attribute</value>
            <value>dimension</value>
            <value>measure</value>
            <value>filter</value>
        </supportedObjects>
        <subQueriesSupported>true</subQueriesSupported>
        <resultObjectInFilterSupported>true</resultObjectInFilterSupported>
        <objectComparisonSupported>true</objectComparisonSupported>
        <constantComparisonSupported>true</constantComparisonSupported>
    </filterCapabilities>
    <ResultObjectCapabilities>
        <useAttributeSeparatelyAvailable>true</useAttributeSeparatelyAvailable>
        <sortObjectsAvailable>true</sortObjectsAvailable>
        <sortNonResultObjectsAvailable>false</sortNonResultObjectsAvailable>
    </ResultObjectCapabilities>
</capabilities>
```

Related Information

Getting the List of the Universes (Semantic Layer) [page 166]

7.1.5 Getting the Link Groups of a Universe

Usage

Gets the link groups of a UNX universe published to a CMS repository.

> **Note**
>
> Link groups of UNV universes are not supported in this release.

About Link Groups

In a relational universe, a link group is a set of business objects sharing a dependency link that is expressed in the universe data source model (data foundation) or the universe semantic layer model (business layer). A link group can be one of the following:

- A data link group
  It is based on the data foundation and it expresses relationships between objects, such as a dimension and another dimension, or a dimension and a measure. A data link group can only contain dimensions and measures. A data link group can be seen as a measure group if it contains at least one measure.
- A semantic link group
  It is based on the business layer and it expresses relationships between objects, such as a dimension and an attribute. A semantic link group must contain one dimension and its associated attributes if any.

A link group cannot be included into another link group. Link groups can have common dimensions or measures.

A universe contains at least one link group, otherwise it is empty.

Request

`GET /universes/<universeID>/linkgroups`

Request type: `application/xml`

Response

Response type: `application/xml`

Response body:

```
<linkGroups>
    <linkGroup type="semantic|data">
        <link name="string" id="string"/>
```

Where:

- The possible values of the link group type are:
  - `data`, if the link group is based on the data foundation
  - `semantic`, if it is based on the business layer
- `name` is the business object path
- `ID` is the identifier of the business object identifier

Example

```
<linkGroups xmlns="http://www.sap.com/rws/sl/universe">
    <linkGroup type="semantic">
        <link name="Customer|folder\Customer|dimension" id="OBJ_13"/>
        <link name="Customer|folder\Customer|dimension\Age|attribute"
id="OBJ_9"/>
    </linkGroup>
    <linkGroup type="data">
        <link name="Resort|folder\Service|dimension" id="OBJ_5"/>
```

```
        <link name="Sales|folder\Sales Date|dimension" id="OBJ_23"/>
        <link name="Resort|folder\Service Line|dimension" id="OBJ_4"/>
        <link name="Customer|folder\Customer|dimension" id="OBJ_13"/>
        <link name="Resort|folder\Resort|dimension" id="OBJ_2"/>
        <link name="Resort|folder\Resort Country|dimension" id="OBJ_6"/>
        <link name="Customer|folder\Customer Country|dimension" id="OBJ_17"/>
        <link name="Measures|folder\Revenue|measure" id="OBJ_7"/>
        <link name="Measures|folder\Number of guests|measure" id="OBJ_20"/>
    </linkGroup>
    <linkGroup type="data">
        <link name="Resort|folder\Service|dimension" id="OBJ_5"/>
        <link name="Reservations|folder\Reservation Date|dimension" id="OBJ_30"/>
        <link name="Resort|folder\Service Line|dimension" id="OBJ_4"/>
        <link name="Customer|folder\Customer|dimension" id="OBJ_13"/>
        <link name="Resort|folder\Resort|dimension" id="OBJ_2"/>
        <link name="Resort|folder\Resort Country|dimension" id="OBJ_6"/>
        <link name="Customer|folder\Customer Country|dimension" id="OBJ_17"/>
        <link name="Measures|folder\Future guests|measure" id="OBJ_37"/>
    </linkGroup>
</linkGroups>
```

Related Information

Getting the List of the Universes (Semantic Layer) [page 166]

7.1.6 Getting the Object Parameters of a Universe

Usage

Gets the list of object parameters used to prompt some values to the end-user and defined in data foundations and business layers of a UNX universe stored in the CMS repository.

> **Note**
>
> This only relates to UNX universes since UNV universes do not support object parameters.

Request

```
GET /universes/<universeID>/prompts
```

Response

Response type: `application/xml` or `application/json`

Response body: the list of the object parameters (`<prompt>`) identified by the following elements:

- `<id>`
- `<dataType>`
- `<name>`
- `<question>`

Example

`GET /universes/6660/prompts`

```
<prompts>
    <prompt>
        <id>_HDBaULaYEeCH2d6gof1MOA</id>
        <dataType>Numeric</dataType>
        <name>Reference Quarter Number</name>
        <question>Reference Quarter Number:</question>
    </prompt>
    <prompt>
        <id>_PImaULaYEeCH2d6gof1MOA</id>
        <dataType>Numeric</dataType>
        <name>Reference Year Number</name>
        <question>Reference Year Number:</question>
    </prompt>
    <prompt>
        <id>_capysLIYEeCVLNpRXueEkA</id>
        <dataType>String</dataType>
        <name>Period Type</name>
        <question>Period Type:</question>
    </prompt>
    <prompt>
        <id>_lBRDMLIYEeCVLNpRXueEkA</id>
        <dataType>String</dataType>
        <name>Time Window</name>
        <question>Time Window:</question>
    </prompt>
    <prompt>
        <id>_uPlz8LIYEeCVLNpRXueEkA</id>
        <dataType>Date</dataType>
        <name>Reference Date</name>
        <question>Reference Date:</question>
    </prompt>
</prompts>
```

7.2 Managing Queries

The BI Semantic Layer RESTful Web Service SDK provides a series of APIs to query UNV and UNX universes and get data. Universe data is exposed as an OData service.

Creating a Query [page 180]

Getting the List of Queries [page 181]

Getting the Details of a Query [page 182]

Deleting a Query [page 183]

7.2.1 Creating a Query

Usage

Creates a precompiled query statement in an instance of the open session.

This instance can be used to run this statement multiple times. The query is not stored to the CMS repository, but in memory.

Request

`POST /queries`

Request type: `application/xml`

Request body: see query specification body schema [page 105]

Response

Response type: `application/xml`

The response is a message stating the success or failure of the request.

Example: Creating a Query with Result Objects Only

`POST /queries`

Request body:

```
<query dataSourceType="unx" dataSourceId="5808" xmlns="http://www.sap.com/rws/sl/
universe">
    <querySpecification version="1.0">
        <queryData>
            <resultObjects>
                <resultObject path="Customer|folder\Geography|folder\City|
dimension" id="_IBo8L7IhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Customer|folder\Customer|dimension"
id="_IBo8OrIhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Inventory|folder\Stock Level|measure"
id="_IB8eFrIhEeCk0Ylv-tlF2Q"/>
            </resultObjects>
        </queryData>
    </querySpecification>
</query>
```

Response body:

```
<success>
```

```
    <message>The resource of type "query" with identifier "6089913651317040730"
has been successfully created.</message>
    <id>6089913651317040730</id>
</success>
```

The response contains the query identifier as `<id>`.

Related Information

Deleting a Query [page 183]

7.2.2 Getting the List of Queries

Usage

Gets the list of all query statements available in the instance of the open session.

Request

`GET /queries`

Request type: `application/xml`

Response

Response type: `application/xml`

Response body: a series of `<query>` elements with the following attributes:

- `dataSourceType` is the universe type (`unv` or `unx`)
- `dataSourceId` is the universe identifier

A `<query>` has an `<id>` child element that represents the corresponding query statement identifier.

Example

`GET /queries`

```
<queries xmlns="http://www.sap.com/rws/sl/universe">
    <query dataSourceType="unx" dataSourceId="5845">
        <id>5602099021259262832</id>
```

```
    </query>
    <query dataSourceType="unx" dataSourceId="5845">
        <id>7059150136676433395</id>
    </query>
</queries>
```

7.2.3 Getting the Details of a Query

Usage

Returns the query specification of a query statement created in an instance of the open session.

Request

`GET /queries/<queryID>`

Request type: `application/xml`

Response

Response type: `application/xml`

Response body: see query specification body schema [page 105]

Example

`GET /queries/6089913651317040730`

```
<query xmlns="http://www.sap.com/rws/sl/universe" id="6089913651317040730"
dataSourceType="unx" dataSourceId="5808">
    <querySpecification version="1.0">
        <queryData>
            <resultObjects>
                <resultObject path="Customer|folder\Geography|folder\City|
dimension" id="_IBo8L7IhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Customer|folder\Customer|dimension"
id="_IBo8OrIhEeCk0Ylv-tlF2Q"/>
                <resultObject path="Inventory|folder\Stock Level|measure"
id="_IB8eFrIhEeCk0Ylv-tlF2Q"/>
            </resultObjects>
        </queryData>
    </querySpecification>
</query>
```

Related Information

Getting the List of Queries [page 181]

7.2.4 Deleting a Query

Usage

Deletes a query statement from memory in the instance of the open session.

> ⓘ Note
>
> You can delete all query statements created in this instance by closing the session.

Request

`DELETE /queries/<queryID>`

Request type: `application/xml`

Response

Response type: `application/xml`

The response is a message stating the success or failure of the request.

Example

`DELETE /queries/6089913651317040730`

```
<success>
    <message>The resource of type "query" with identifier "6089913651317040730"
has been successfully removed.</message>
    <id>6089913651317040730</id>
</success>
```

Related Information

Getting the List of Queries [page 181]

7.3 Managing Parameters

The BI Semantic Layer RESTful Web Service SDK provides a series of APIs to perform tasks on contexts and prompts of a UNV or UNX universe.

Getting the List of Parameters [page 184]

Responding to Parameters [page 187]

Getting the Details of a Parameter [page 199]

7.3.1 Getting the List of Parameters

Usage

Returns the first contexts or parameters to be answered.

Request

`GET /queries/<queryID>/parameters?formattedValues=<formattedValues>`

Where:

- `<formattedValues>` is an optional, Boolean parameter. Default value is `false`. If set to `true`, the DateTime and Numeric values are formatted according to the preferred viewing locale set through `X-SAP-PVL` in the request.

Response

Response type: `application/xml` or `application/json`

The response provides the parameters with their expected answers and default values. See Parameter Response Body Schemas [page 133] to learn about the content structure and element details.

Example: No parameter

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<parameters/>
```

Example: Context

The parameter of type `context` provides two possible context values.

```
<parameters>
    <parameter type="context" optional="false" dpId="DP0" >
        <id>0</id>
        <name>Select a context</name>
        <answer type="text" constrained="true">
            <info cardinality="Single">
                <lov partial="false">
                    <values>
                        <value id="C2">Reservations</value>
                        <value id="C1">Sales</value>
                    </values>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

Example: DateTime Prompt

The parameter of type `prompt` accepts only one answer value (cardinality `Single`).

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>0</id>
        <name>Enter Open Date:</name>
        <answer constrained="false" type="DateTime">
            <info cardinality="Single">
                <lov refreshable="true" partial="false" hierarchical="false">
                    <values>
                        <value>2011-09-03T17:15:00.000+02:00</value>  <!--
possible values -->
                        <value>2012-09-03T17:15:00.000+02:00</value>
                        <value>2013-09-03T17:15:00.000+02:00</value>
                    </values>
                </lov>
                <values>
                    <value>2011-09-03T17:15:00.000+02:00</value>  <!-- default
value -->
                </values>
            </info>
            <values>
                <value>2011-09-03T17:15:00.000+02:00</value> <!-- default value
-->
            </values>
        </answer>
    </parameter>
</parameters>
```

Example: Prompt with Multiple Values

The parameter of type `prompt` accepts multiple answer values (cardinality `Multiple`).

```
<parameters>
    <parameter dpId="DP0" type="prompt" optional="false">
        <id>0</id>
        <technicalName>pmmultiple</technicalName>
        <name>Enter Cityname (multiple)</name>
        <answer type="Text" constrained="true">
            <info cardinality="Multiple">
                <lov refreshable="true" partial="false" hierarchical="false">
                    <id>UNIVERSELOV_DS0.DO22</id>
                    <values>
                        <value>Barcelona</value>  <!-- possible values -->
                        <value>Berlin</value>
                        <value>Copenhagen</value>
                        <value>Firenze</value>
                        <value>London</value>
                        <value>Madrid</value>
                        <value>Marseilles</value>
                        <value>Oslo</value>
                        <value>Paris</value>
                        <value>Prague s</value>
                        <value>Roma</value>
                        <value>Stockholm</value>
                        <value>Stuttgart</value>
                    </values>
                    <columns mappingId="0">
                        <column type="String" id="0">Cityname</column>
                    </columns>
                </lov>
                <values>
                    <value>Barcelona</value>  <!-- default values -->
                    <value>Berlin</value>
                </values>
            </info>
            <values>
                <value>Barcelona</value> <!-- default values -->
                <value>Berlin</value>
            </values>
        </answer>
    </parameter>
</parameters>
```

Related Information

Parameter Response Body Schemas [page 133]
Getting the List of Queries [page 181]

7.3.2 Responding to Parameters

Usage

Responds to contexts and prompts.

Returns additional parameters if found when the first parameters have been responded. Since these new parameters must also be responded, you must perform an additional call by using as request body the answers to the new parameters, and the answers to the parameters that have triggered the new parameters.

Request

`PUT /queries/<queryID>/parameters?formattedValues=<formattedValues>`

Where:

- `<formattedValues>` is an optional, Boolean parameter. Default value is `false`. If set to `true`, the DateTime and Numeric values are formatted according to the preferred viewing locale set through `X-SAP-PVL` in the request.

Request body: the answers to the parameters retrieved using the `GET .../parameters` call. See Answer Request Body Schemas [page 142] to learn about the XML content structure and element details.

Response

Response type: `application/xml` or `application/json`

When all parameters have been answered, the last `PUT` call returns a message stating the success of the request.

```
<success>
    <message>The parameters of the resource of type "Query" with identifier "XX"
have been answered.</message>
    <id>XX</id>
</success>
```

Example - Responding to a Context [page 188]

Example - Responding to a Prompt with Multiple Columns [page 189]

Example - Responding to two Prompts that Define an Interval [page 190]

Example - Responding to Cascading Parameters [page 192]

Example - Responding to a Hierarchical Parameter [page 196]

Related Information

Getting the List of Queries [page 181]

7.3.2.1 Example - Responding to a Context

A `GET .../parameters` call returns a list of values for a specific parameter of type `context`:

```
<parameters>
    <parameter dpId="DP0" type="context" optional="false">
        <id>0</id>
        <technicalName>c _Sub_2</technicalName>
        <name>Select a context</name>
        <answer type="Text" constrained="true">
            <info cardinality="Single">
                <lov partial="false">
                    <values>
                        <value id="CTX_2">Reservations</value>
                        <value id="CTX_1">Sales</value>
                    </values>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

In the `PUT` call, the request body contains an answer to the context. The following request body does not show the `<info>` element, which is not mandatory:

```
<parameters>
    <parameter dpId="DP0" optional="false" type="context">
        <id>0</id>
        <technicalName>c _Sub_2</technicalName>
        <name>Select a context</name>
        <answer constrained="true" type="Text">
            <values>
                <value id="CTX_2">Reservations</value>
            </values>
        </answer>
    </parameter>
</parameters>
```

The `PUT` call returns a successful message.

```
<success>
    <message>The resource of type "Query" with identifier "XX" has not been
modified.</message>
    <id>XX</id>
</success>
```

7.3.2.2 Example - Responding to a Prompt with Multiple Columns

The `GET .../parameters` call returns an XML that describes a parameter of type `prompt` and cardinality `Single` that accepts an answer made of two column values. The `Employee Name` column is the one used as reference. The default value is `Clive Lord`.

```
<parameters>
    <parameter dpId="DP0" optional="false" type="prompt">
        <id>1</id>
        <technicalName>psCustomLOV_2col</technicalName>
        <name>Enter CustomLOV_2col:</name>
        <answer constrained="false" type="Text">
            <info cardinality="Single">
                <lov hierarchical="false" partial="false" refreshable="true">
                    <id>UNIVERSELOV_DS0.DO65</id>
                    <cvalues>
                        <cvalue>
                            <column id="0">Alan Leroy</column>  <!-- possible
values -->
                            <column id="1">Sales</column>
                        </cvalue>
                        <cvalue>
                            <column id="0">Brian Lunn</column>
                            <column id="1">Board</column>
                        </cvalue>
                        <cvalue>
                            <column id="0">Brian Opel</column>
                            <column id="1">Sales</column>
                        </cvalue>
                        <cvalue>
                            <column id="0">Chris Hutchings</column>
                            <column id="1">Sales</column>
                        </cvalue>
                        <cvalue>
                            <column id="0">Clive Allen</column>
                            <column id="1">Support</column>
                        </cvalue>
                        <cvalue>
                            <column id="0">Clive Lord</column>
                            <column id="1">Sales</column>
                        </cvalue>
                        ...
                        <cvalue>
                            <column id="0">Yvonne Viamont</column>
                            <column id="1">Support</column>
                        </cvalue>
                    </cvalues>
                    <columns mappingId="0">
                        <column id="0" type="String">Employee Name</column>
                        <column id="1" type="String">Department</column>
                    </columns>
                </lov>
                <values>
                    <value>Clive Lord</value>  <!-- default value -->
                </values>
            </info>
            <values>
                <value>Clive Lord</value>  <!-- default value -->
            </values>
        </answer>
    </parameter>
</parameters>
```

The `PUT .../parameters` call responds to the parameter. The following request body does not show the `<info>` element, which is not mandatory. The answer value is the default value.

```
<parameters>
    <parameter dpId="DP0" optional="false" type="prompt">
        <id>1</id>
        <technicalName>psCustomLOV_2col</technicalName>
        <name>Enter CustomLOV_2col:</name>
        <answer constrained="false" type="Text">
            <values>
                <value>Clive Lord</value>
            </values>
        </answer>
    </parameter>
</parameters>
```

Response:

```
<success>
    <message>The resource of type "Query" with identifier "5661410011133153584"
has not been modified.</message>
    <id>5661410011133153584</id>
</success>
```

7.3.2.3 Example - Responding to two Prompts that Define an Interval

The `GET .../parameters` call returns an XML that describes two parameters of type `prompt` and cardinality `Single` that define a journey between two European cities:

```
<parameters>
    <parameter dpId="DP0" type="prompt" optional="false">
        <id>0</id>
        <technicalName>psEnter City of Departure</technicalName>
        <name>Enter City of Departure</name>
        <answer type="Text" constrained="true">
            <info cardinality="Single">
                <lov refreshable="true" partial="false" hierarchical="false">
                    <id>UNIVERSELOV_DS0.DO22</id>
                    <values>
                        <value>Barcelona</value>  <!-- possible values -->
                        <value>Berlin</value>
                        <value>Copenhagen</value>
                        <value>Firenze</value>
                        <value>London</value>
                        <value>Madrid</value>
                        <value>Marseilles</value>
                        <value>Oslo</value>
                        <value>Paris</value>
                        <value>Prague</value>
                        <value>Roma</value>
                        <value>Stockholm</value>
                        <value>Stuttgart</value>
                    </values>
                    <columns mappingId="0">
                        <column type="String" id="0">Cityname</column>
                    </columns>
                </lov>
            </info>
        </answer>
```

```
    </parameter>
    <parameter dpId="DP0" type="prompt" optional="false">
        <id>1</id>
        <technicalName>psEnter Destination City</technicalName>
        <name>Enter Destination City</name>
        <answer type="Text" constrained="true">
            <info cardinality="Single">
                <lov refreshable="true" partial="false" hierarchical="false">
                    <id>UNIVERSELOV_DS0.DO22</id>
                    <values>
                        <value>Barcelona</value>  <!-- possible values -->
                        <value>Berlin</value>
                        <value>Copenhagen</value>
                        <value>Firenze</value>
                        <value>London</value>
                        <value>Madrid</value>
                        <value>Marseilles</value>
                        <value>Oslo</value>
                        <value>Paris</value>
                        <value>Prague s</value>
                        <value>Roma</value>
                        <value>Stockholm</value>
                        <value>Stuttgart</value>
                    </values>
                    <columns mappingId="0">
                        <column type="String" id="0">Cityname</column>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

The `PUT .../parameters` call responds to the prompts with the following request body:

```
<parameters>
    <parameter dpId="DP0" optional="false" type="prompt">
        <id>0</id>
        <technicalName>psEnter City of Departure</technicalName>
        <name>Enter City of Departure</name>
        <answer constrained="true" type="Text">
            <values>
                <value>Barcelona</value>
            </values>
        </answer>
    </parameter>
    <parameter dpId="DP0" optional="false" type="prompt">
        <id>1</id>
        <technicalName>psEnter Destination City</technicalName>
        <name>Enter Destination City</name>
        <answer constrained="true" type="Text">
            <values>
                <value>Berlin</value>
            </values>
        </answer>
    </parameter>
</parameters>
```

Response:

```
<success>
    <message>The resource of type "Query" with identifier "7063116525765987187"
has not been modified.</message>
    <id>7063116525765987187</id>
</success>
```

7.3.2.4 Example - Responding to Cascading Parameters

The `GET .../parameters` call returns:

- The list of possible values for the answer to a parameter of `id=0`, type `prompt`, and cardinality `Multiple`
- A second parameter of `id=1`, type `prompt`, and cardinality `Single`

The first parameter needs to be answered so that you can get the list of values of the second parameter.

```
<parameters>
    <parameter dpId="DP0" type="prompt" optional="false">
        <id>0</id>
        <technicalName>pmEnter First Name of Customer</technicalName>
        <name>Enter First Name of Customer</name>
        <answer type="Text" constrained="false">
            <info cardinality="Multiple">
                <lov refreshable="true" partial="false" hierarchical="false">
                    <id>UNIVERSELOV_DS0.DO5a</id>
                    <values>
                        <value>Andre</value>
                        <value>Annie</value>
                        <value>Brian</value>
                        <value>Christine</value>
                        <value>Dan</value>
                        <value>Erika</value>
                        <value>Franz</value>
                        <value>George</value>
                        <value>Habib</value>
                        <value>Jack</value>
                        <value>Jan</value>
                        <value>Joe</value>
                        <value>John</value>
                        <value>Luke</value>
                        <value>Mary</value>
                        <value>Michele</value>
                        <value>Paul</value>
                        <value>Peter</value>
                        <value>Philip</value>
                        <value>Robert</value>
                        <value>Robin</value>
                    </values>
                    <columns mappingId="0">
                        <column type="String" id="0">First Name</column>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
    <parameter dpId="DP0" type="prompt" optional="false">
        <id>1</id>
        <technicalName>psInvoice Date</technicalName>
        <name>Enter Invoice Date:</name>
        <answer type="Date" constrained="false">
            <info cardinality="Single">
                <lov refreshable="true" hierarchical="false">
                    <id>UNIVERSELOV_DS0.DO6b</id>
                    <parameters>
                        <id>0</id>
                    </parameters>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

You answer the parameter with the `PUT .../parameters` call. First, you query the list of values according to a certain pattern (`a*`).

Request body:

```
<parameters>
    <parameter dpId="DP0" optional="false" type="prompt">
        <id>0</id>
        <technicalName>pmEnter First Name of Customer</technicalName>
        <name>Enter First Name of Customer</name>
        <answer constrained="false" type="Text">
            <info cardinality="Multiple">
                <lov hierarchical="false" partial="false" refreshable="true">
                    <query>
                        <sort order="Descending"/>
                        <search>a*</search>
                    </query>
                </lov>
            </info>
        </answer>
    </parameter>
    <parameter dpId="DP0" optional="false" type="prompt">
        <id>1</id>
        <technicalName>psInvoice Date</technicalName>
        <name>Enter Invoice Date:</name>
        <answer constrained="false" type="Date">
            <info cardinality="Single">
                <lov hierarchical="false" refreshable="true">
                    <query>
                        <sort order="Descending"/>
                    </query>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

The `PUT` call returns the possible values for the parameter of `id=0` that correspond to the search pattern in the descending order.

Response:

```
<parameters>
    <parameter dpId="DP0" type="prompt" optional="false">
        <id>0</id>
        <technicalName>pmEnter First Name of Customer</technicalName>
        <name>Enter First Name of Customer</name>
        <answer type="Text" constrained="false">
            <info cardinality="Multiple">
                <lov refreshable="true" partial="false" hierarchical="false">
                    <id>UNIVERSELOV_DS0.DO5a</id>
                    <values>
                        <value>Annie</value>
                        <value>Andre</value>
                    </values>
                    <columns mappingId="0">
                        <column type="String" id="0">First Name </column>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
    <parameter dpId="DP0" type="prompt" optional="false">
        <id>1</id>
        <technicalName>psInvoice Date</technicalName>
        <name>Enter Invoice Date:</name>
```

```
        <answer type="Date" constrained="false">
            <info cardinality="Single">
                <lov refreshable="true" hierarchical="false">
                    <id>UNIVERSELOV_DS0.DO6b</id>
                    <parameters>
                        <id>0</id>
                    </parameters>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

Another `PUT` call with the values for the parameter of `id=0` as input is needed to get the values for the second parameter of `id=1`.

Request body:

```
<parameters>
    <parameter dpId="DP0" optional="false" type="prompt">
        <id>0</id>
        <technicalName>pmEnter First Name of Customer</technicalName>
        <name>Enter First Name of Customer</name>
        <answer constrained="false" type="Text">
            <info cardinality="Multiple">
                <lov hierarchical="false" partial="false" refreshable="true">
                    <query>
                        <sort order="Descending"/>
                        <search>a*</search>
                    </query>
                </lov>
            </info>
            <values>
                <value>Andre</value>
                <value>Annie</value>
            </values>
        </answer>
    </parameter>
    <parameter dpId="DP0" optional="false" type="prompt">
        <id>1</id>
        <technicalName>psInvoice Date</technicalName>
        <name>Enter Invoice Date:</name>
        <answer constrained="false" type="Date">
            <info cardinality="Single">
                <lov hierarchical="false" refreshable="true">
                    <query>
                        <sort order="Descending"/>
                    </query>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

Response:

```
<parameters>
    <parameter dpId="DP0" type="prompt" optional="false">
        <id>1</id>
        <technicalName>psInvoice Date</technicalName>
        <name>Enter Invoice Date:</name>
        <answer type="Date" constrained="false">
            <info cardinality="Single">
                <lov refreshable="true" partial="false" hierarchical="false">
                    <id>UNIVERSELOV_DS0.DO6b</id>
                    <values>
```

```
                        <value>1995-09-16T17:04:00.000+02:00</value>
                        <value>1994-12-24T07:47:00.000+01:00</value>
                        <value>1994-12-14T16:26:00.000+01:00</value>
                        <value>1994-07-24T05:55:00.000+02:00</value>
                        <value>1994-04-06T12:43:00.000+02:00</value>
                        <value>1993-12-13T10:05:00.000+01:00</value>
                        <value>1993-12-01T15:54:00.000+01:00</value>
                        <value>1993-10-07T00:16:00.000+01:00</value>
                        <value>1993-07-08T00:12:00.000+02:00</value>
                        <value>1993-05-22T02:00:00.000+02:00</value>
                        <value>1993-05-12T02:35:00.000+02:00</value>
                        <value>1993-01-24T13:55:00.000+01:00</value>
                        <value>1992-11-28T15:26:00.000+01:00</value>
                        <value>1992-11-24T19:05:00.000+01:00</value>
                        <value>1992-10-07T04:27:00.000+01:00</value>
                    </values>
                    <columns mappingId="0">
                        <column type="DateTime" id="0">Invoice Date </column>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```


The last `PUT` call with the value chosen for the parameter of `id=1` as input returns a successful message.

Request body:


```
<parameters>
    <parameter dpId="DP0" optional="false" type="prompt">
        <id>0</id>
        <technicalName>pmEnter First Name of Customer</technicalName>
        <name>Enter First Name of Customer</name>
        <answer constrained="false" type="Text">
            <info cardinality="Multiple">
                <lov hierarchical="false" partial="false" refreshable="true">
                    <query>
                        <sort order="Descending"/>
                        <search>a*</search>
                    </query>
                </lov>
            </info>
            <values>
                <value>Andre</value>
                <value>Annie</value>
            </values>
        </answer>
    </parameter>
    <parameter dpId="DP0" optional="false" type="prompt">
        <id>1</id>
        <technicalName>psInvoice Date</technicalName>
        <name>Enter Invoice Date:</name>
        <answer constrained="false" type="Date">
            <info cardinality="Single">
                <lov hierarchical="false" refreshable="true">
                    <query>
                        <sort order="Descending"/>
                    </query>
                </lov>
            </info>
            <values>
                <value>1993-05-22T02:00:00.000+02:00</value>
            </values>
        </answer>
    </parameter>
</parameters>
```

Response:

```
<success>
    <message>The resource of type "Query" with identifier "6366177434639509348"
has not been modified.</message>
    <id>6366177434639509348</id>
</success>
```

7.3.2.5 Example - Responding to a Hierarchical Parameter

A `GET .../parameters` call returns a list of values for a specific parameter of type `prompt` that asks you to select a city. Values are continents.

Response:

```
<parameters>
    <parameter dpId="DP0" type="prompt" optional="false">
        <id>0</id>
        <technicalName>psSelect a city</technicalName>
        <name>Select a city for customer</name>
        <answer type="Text" constrained="true">
            <info cardinality="Single">
                <lov refreshable="true" partial="false" hierarchical="true">
                    <id>UNIVERSELOV_DS0.DO6a</id>
                    <updated>1970-01-01T01:00:00.000+01:00</updated>
                    <values>
                        <value final="false">Asia</value>
                        <value final="false">Europe</value>
                        <value final="false">Latin America</value>
                        <value final="false">North America</value>
                    </values>
                    <columns mappingId="0">
                        <column type="String" id="0"/>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

In the first `PUT` call, the value of the first level of the list of values is given as answer to the parameter. The answer is `Europe`.

Request body:

```
<parameters>
    <parameter dpId="DP0" optional="false" type="prompt">
        <id>0</id>
        <technicalName>psSelect a city</technicalName>
        <name>Select a city for customer</name>
        <answer constrained="true" type="Text">
            <info cardinality="Single">
                <lov hierarchical="true" partial="false" refreshable="true">
                    <query>
                        <path>
                            <value>Europe</value>
                        </path>
                    </query>
                </lov>
            </info>
        </answer>
```

```
    </parameter>
</parameters>
```

The response contains the list of values of the second level of the parameter. Values represent countries.

Response:

```
<parameters>
    <parameter dpId="DP0" type="prompt" optional="false">
        <id>0</id>
        <technicalName>psSelect a city</technicalName>
        <name>Select a city for customer</name>
        <answer type="Text" constrained="true">
            <info cardinality="Single">
                <lov refreshable="true" partial="false" hierarchical="true"
path="[[0,\ Europe]]">
                    <id>UNIVERSELOV_DS0.DO6a</id>
                    <updated>1970-01-01T01:00:00.000+01:00</updated>
                    <values>
                        <value final="false">France</value>
                        <value final="false">Germany</value>
                        <value final="false">Italy</value>
                        <value final="false">Spain</value>
                        <value final="false">United-Kingdom</value>
                    </values>
                    <columns mappingId="0">
                        <column type="String" id="0"/>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

In a second `PUT` call, the value of the list of values of the second level is given as answer to the parameter (`France`). The first level value is recalled (`Europe`).

Request body:

```
<parameters>
    <parameter dpId="DP0" optional="false" type="prompt">
        <id>0</id>
        <technicalName>psSelect a city</technicalName>
        <name>Select a city for customer</name>
        <answer constrained="true" type="Text">
            <info cardinality="Single">
                <lov hierarchical="true" partial="false" refreshable="true">
                    <query>
                        <path>
                            <value>Europe</value>
                            <value>France</value>
                        </path>
                    </query>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

The response contains the list of values of the third and last level of the parameter. Values represent cities.

Response:

```
<parameters>
    <parameter dpId="DP0" type="prompt" optional="false">
```

```
        <id>0</id>
        <technicalName>psSelect a city</technicalName>
        <name>Select a city for customer</name>
        <answer type="Text" constrained="true">
            <info cardinality="Single">
                <lov refreshable="true" partial="false" hierarchical="true"
path="[[0,\ Europe], [0,\ France]]">
                    <id>UNIVERSELOV_DS0.DO6a</id>
                    <updated>1970-01-01T01:00:00.000+01:00</updated>
                    <values>
                        <value>Marseilles</value>
                        <value>Paris</value>
                    </values>
                    <columns mappingId="0">
                        <column type="String" id="0"/>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

In the third and last PUT call, values of the lists of values of the first and second levels, and values of the third level are given as answer to the parameter. Only the value of the third level is considered as actual parameter value (`Paris`).

Request body:

```
<parameters>
    <parameter dpId="DP0" optional="false" type="prompt">
        <id>0</id>
        <technicalName>psSelect a city</technicalName>
        <name>Select a city for customer</name>
        <answer constrained="true" type="Text">
            <info cardinality="Single">
                <lov hierarchical="true" partial="false" refreshable="true">
                    <query>
                        <path>
                            <value>Europe</value>
                            <value>France</value>
                        </path>
                    </query>
                </lov>
            </info>
            <values>
                <value path="[[0,\ Europe],[0,\ France]]">Paris</value>
            </values>
        </answer>
    </parameter>
</parameters>
```

Response:

```
<success>
    <message>The resource of type "Query" with identifier "6512967723055172061"
has not been modified.</message>
    <id>6512967723055172061</id>
</success>
```

7.3.3 Getting the Details of a Parameter

Usage

Returns the possible answers of a parameter specified by its identifier.

Request

`GET /queries/<queryID>/parameters/<parameterID>?formattedValues=<formattedValues>`

Where:

- `<formattedValues>` is an optional, Boolean parameter. Default value is `false`. If set to `true`, the DateTime and Numeric values are formatted according to the preferred viewing locale set through `X-SAP-PVL` in the request.

Response

Response type: `application/xml` or `application/json`

The response provides the parameter with its expected answers. See Parameter Response Body Schemas [page 133] to learn about the content structure and element details.

Example: Parameter of Type Prompt

`GET /queries/8963412105615728541/parameters/0`

Response:

```
<parameters>
    <parameter type="prompt" optional="false">
        <id>0</id>
        <technicalName>psEnd</technicalName>
        <name>Enter value for Customer Id (End):</name>
        <answer type="Numeric" constrained="true">
            <info keepLastValues="true" cardinality="Single">
                <lov mandatorySearch="false" searchable="true"
refreshable="true" partial="false" hierarchical="false">
                    <id>UNIVERSELOV_DS0.DOa8</id>
                    <updated>2015-08-19T17:18:53.000+02:00</updated>
                    <intervals>
                        <interval id="0">
                            <cvalue>
                                <column id="0">1991</column>
                                <column id="1">2002-02-09T01:00:00.000+01:00</
column>
                                <column id="2">2002-04-10T02:00:00.000+02:00</
column>
```

```
                            </cvalue>
                            <cvalue>
                                <column id="0">1616</column>
                                <column id="1">2001-06-15T02:00:00.000+02:00</
column>
                                <column id="2">2001-08-14T02:00:00.000+02:00</
column>
                            </cvalue>
                        </interval>
                        <interval id="1">
                            <cvalue>
                                <column id="0">1614</column>
                                <column id="1">2001-11-10T01:00:00.000+01:00</
column>
                                <column id="2">2002-01-09T01:00:00.000+01:00</
column>
                            </cvalue>
                            <cvalue>
                                <column id="0">1139</column>
                                <column id="1">2001-09-03T02:00:00.000+02:00</
column>
                                <column id="2">2001-11-02T01:00:00.000+01:00</
column>
                            </cvalue>
                        </interval>
                        <interval id="2">
                            <cvalue>
                                <column id="0">1132</column>
                                <column id="1">2001-12-14T01:00:00.000+01:00</
column>
                                <column id="2">2002-02-12T01:00:00.000+01:00</
column>
                            </cvalue>
                            <cvalue>
                                <column id="0">718</column>
                                <column id="1">2002-02-09T01:00:00.000+01:00</
column>
                                <column id="2">2002-04-10T02:00:00.000+02:00</
column>
                            </cvalue>
                        </interval>
                        <interval id="3">
                            <cvalue>
                                <column id="0">714</column>
                                <column id="1">2001-12-28T01:00:00.000+01:00</
column>
                                <column id="2">2002-02-26T01:00:00.000+01:00</
column>
                            </cvalue>
                            <cvalue>
                                <column id="0">297</column>
                                <column id="1">2001-10-08T02:00:00.000+02:00</
column>
                                <column id="2">2001-12-07T01:00:00.000+01:00</
column>
                            </cvalue>
                        </interval>
                        <interval id="4">
                            <cvalue>
                                <column id="0">293</column>
                                <column id="1">2001-12-16T01:00:00.000+01:00</
column>
                                <column id="2">2002-02-14T01:00:00.000+01:00</
column>
                            </cvalue>
                            <cvalue>
                                <column id="0">1</column>
```

```
                                <column id="1">2001-08-09T02:00:00.000+02:00</
column>
                                <column id="2">2001-10-08T02:00:00.000+02:00</
column>
                            </cvalue>
                        </interval>
                    </intervals>
                    <cvalues>
                        <cvalue>
                            <column id="0">1991</column>
                            <column id="1">2002-02-09T01:00:00.000+01:00</column>
                            <column id="2">2002-04-10T02:00:00.000+02:00</column>
                        </cvalue>
                        <cvalue>
                            <column id="0">1988</column>
                            <column id="1">2001-07-14T02:00:00.000+02:00</column>
                            <column id="2">2001-09-12T02:00:00.000+02:00</column>
                        </cvalue>
                        ...
                    </cvalues>
                    <columns mappingId="0">
                        <column type="Numeric" id="0">Customer Id </column>
                        <column type="DateTime" id="1">From Date </column>
                        <column type="DateTime" id="2">To Date </column>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

Getting the Details of a Parameter Restricted Through a Query

Usage

Returns the possible answers of a parameter of which values are restricted through a query.

Request

`PUT /queries/<queryID>/parameters/<parameterID>?formattedValues=<formattedValues>`

Where:

- `<formattedValues>` is an optional, Boolean parameter. Default value is `false`. If set to `true`, the DateTime and Numeric values are formatted according to the preferred viewing locale set through `X-SAP-PVL` in the request.

Request body: the details of a query. See Answer Request Body Schemas [page 142] to learn about the content structure and element details.

Response

Response type: `application/xml` or `application/json`

The response provides the parameter with its expected answers.

Example: Parameter of Type Prompt

First, you retrieve the `prompt` parameter of cardinality `Single` that accepts an answer made of three columns.

`GET /queries/8963412105615728541/parameters`

Response:


```
<parameters>
    <parameter type="prompt" optional="false">
        <id>0</id>
        <technicalName>psEnd</technicalName>
        <name>Enter value for Customer Id (End):</name>
        <answer type="Numeric" constrained="true">
            <info keepLastValues="true" cardinality="Single">
                <lov mandatorySearch="false" searchable="true"
refreshable="true" partial="false" hierarchical="false">
                    <id>UNIVERSELOV_DS0.DOa8</id>
                    <updated>2015-08-19T17:18:53.000+02:00</updated>
                    <intervals>
                        <interval id="0">
                            <cvalue>
                                <column id="0">1991</column>
                                <column id="1">2002-02-09T01:00:00.000+01:00</
column>
                                <column id="2">2002-04-10T02:00:00.000+02:00</
column>
                            </cvalue>
                            <cvalue>
                                <column id="0">1616</column>
                                <column id="1">2001-06-15T02:00:00.000+02:00</
column>
                                <column id="2">2001-08-14T02:00:00.000+02:00</
column>
                            </cvalue>
                        </interval>
                        <interval id="1">
                            <cvalue>
                                <column id="0">1614</column>
                                <column id="1">2001-11-10T01:00:00.000+01:00</
column>
                                <column id="2">2002-01-09T01:00:00.000+01:00</
column>
                            </cvalue>
                            <cvalue>
                                <column id="0">1139</column>
                                <column id="1">2001-09-03T02:00:00.000+02:00</
column>
                                <column id="2">2001-11-02T01:00:00.000+01:00</
column>
                            </cvalue>
                        </interval>
                        <interval id="2">
                            <cvalue>
                                <column id="0">1132</column>
```

```
                                <column id="1">2001-12-14T01:00:00.000+01:00</
column>
                                <column id="2">2002-02-12T01:00:00.000+01:00</
column>
                            </cvalue>
                            <cvalue>
                                <column id="0">718</column>
                                <column id="1">2002-02-09T01:00:00.000+01:00</
column>
                                <column id="2">2002-04-10T02:00:00.000+02:00</
column>
                           </cvalue>
                        </interval>
                        <interval id="3">
                            <cvalue>
                                <column id="0">714</column>
                                <column id="1">2001-12-28T01:00:00.000+01:00</
column>
                                <column id="2">2002-02-26T01:00:00.000+01:00</
column>
                            </cvalue>
                            <cvalue>
                                <column id="0">297</column>
                                <column id="1">2001-10-08T02:00:00.000+02:00</
column>
                                <column id="2">2001-12-07T01:00:00.000+01:00</
column>
                            </cvalue>
                        </interval>
                        <interval id="4">
                            <cvalue>
                                <column id="0">293</column>
                                <column id="1">2001-12-16T01:00:00.000+01:00</
column>
                                <column id="2">2002-02-14T01:00:00.000+01:00</
column>
                            </cvalue>
                            <cvalue>
                                <column id="0">1</column>
                                <column id="1">2001-08-09T02:00:00.000+02:00</
column>
                                <column id="2">2001-10-08T02:00:00.000+02:00</
column>
                            </cvalue>
                        </interval>
                    </intervals>
                    <cvalues>
                        <cvalue>
                            <column id="0">1991</column>
                            <column id="1">2002-02-09T01:00:00.000+01:00</column>
                            <column id="2">2002-04-10T02:00:00.000+02:00</column>
                        </cvalue>
                        <cvalue>
                            <column id="0">1988</column>
                            <column id="1">2001-07-14T02:00:00.000+02:00</column>
                            <column id="2">2001-09-12T02:00:00.000+02:00</column>
                        </cvalue>
                        ...
                    </cvalues>
                    <columns mappingId="0">
                        <column type="Numeric" id="0">Customer Id </column>
                        <column type="DateTime" id="1">From Date </column>
                        <column type="DateTime" id="2">To Date </column>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
```

```
<parameters>
```

Then you retrieve the possible answer values restricted through a query.

`PUT /queries/8963412105615728541/parameters/0`

Request body:

```
<parameters>
    <parameter optional="false" type="prompt">
        <id>0</id>
        <technicalName>psEnd</technicalName>
        <name>Enter value for Customer Id (End):</name>
        <answer constrained="true" type="Numeric">
            <info cardinality="Single" keepLastValues="true">
                <lov hierarchical="false" mandatorySearch="false"
partial="false" refreshable="true" searchable="true">
                    <id>UNIVERSELOV_DS0.DOa8</id>
                    <updated>2015-08-19T17:18:53.000+02:00</updated>
                    <query>
                        <sort order="Ascending"/>
                    <search>80*</search>
                    </query>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

Response:

```
<parameter type="prompt" optional="false">
    <id>0</id>
    <technicalName>psEnd</technicalName>
    <name>Enter value for Customer Id (End):</name>
    <answer type="Numeric" constrained="true">
        <info keepLastValues="true" cardinality="Single">
            <lov mandatorySearch="false" searchable="true" refreshable="true"
partial="false" hierarchical="false">
                <id>UNIVERSELOV_DS0.DOa8</id>
                <updated>2015-08-19T17:25:38.000+02:00</updated>
                <cvalues>
                    <cvalue>
                        <column id="0">802</column>
                        <column id="1">2001-12-21T01:00:00.000+01:00</column>
                        <column id="2">2002-02-19T01:00:00.000+01:00</column>
                    </cvalue>
                    <cvalue>
                        <column id="0">802</column>
                        <column id="1">2000-08-10T02:00:00.000+02:00</column>
                        <column id="2">2000-10-09T02:00:00.000+02:00</column>
                    </cvalue>
                </cvalues>
                <columns mappingId="0">
                    <column type="Numeric" id="0">Customer Id </column>
                    <column type="DateTime" id="1">From Date </column>
                    <column type="DateTime" id="2">To Date </column>
                </columns>
            </lov>
        </info>
    </answer>
</parameter>
```

7.4 Managing Query Results

After the query has been posted and its parameters resolved, you can run the query and retrieve the resulting data set using some of the BI Semantic Layer RESTful Web Service APIs. The dataset is returned using an OData format. Several calls to OData allow you to retrieve the resulting data set of the query.

Accessing the OData Service [page 205]

Getting the OData Flow Metadata [page 207]

Getting the OData Flow Content [page 209]

OData Content - Getting All Data [page 211]

OData Content - Getting the Row Count [page 214]

OData Content - Getting the First Row [page 215]

OData Content - Getting Property Content [page 215]

OData Content - Getting Property Raw Content [page 216]

OData Content - Getting Content After Offset [page 216]

OData Content - Getting the First N Rows [page 219]

OData Content - Getting the First N Rows After Offset [page 220]

7.4.1 Accessing the OData Service

Usage

Returns the available data flows for the specified query.

Request

```
GET /queries/<queryID>/data.svc
```

Response

Response type: `application/xml` or `application/json`

The response is an Atom feed or a collection of JSON objects.

> ⚠ Caution
>
> The call returns an error message if at least one context or parameter has not been answered.

Example: One Flow

`GET /queries/6089913651317040548/data.svc`

Response type: `application/xml`

```
<service xml:base="http://w2k8x64sp2:6405/biprws/sl/v1/queries/
6089913651317040548/data.svc" xmlns:app="http://www.w3.org/2007/app"
    xmlns:atom="http://www.w3.org/2005/Atom">
    <workspace>
        <atom:title>Default</atom:title>
        <collection href="Flows0">
            <atom:title>Flows0</atom:title>
       </collection>
    </workspace>
</service>
```

Response type: `application/json`

```
{"d":
    {"EntitySets":
        ["Flows0"]
    }
}
```

Example: Multiple Flows

`GET /queries/6089913651317040730/data.svc`

Response type: `application/xml`

```
<service xml:base="http://w2k8x64sp2:6405/biprws/sl/v1/queries/
6089913651317040730/data.svc/" xmlns="http://www.w3.org/2007/app"
    xmlns:atom="http://www.w3.org/2005/Atom">
    <workspace>
        <atom:title>Default</atom:title>
        <collection href="Flows0">
            <atom:title>Flows0</atom:title>
        </collection>
        <collection href="Flows1">
            <atom:title>Flows1</atom:title>
        </collection>
    </workspace>
</service>
```

Response type: `application/json`

```
{"d":
    {"EntitySets":
        ["Flows0","Flows1"]
    }
}
```

Related Information

Getting the List of Queries [page 181]

7.4.2 Getting the OData Flow Metadata

Usage

Gets the metadata used to build the query results that the OData service exposes.

Request

```
GET /queries/<queryID>/data.svc/$metadata
```

Response

Response type: `application/xml`

The response that contains the schema has an EDMX format. The `Id` property does not map any object. It provides the current row number.

Example: Query with Multiple Flows

```
GET sl/v1/queries/9147840914933280297/data.svc/$metadata
```

```
<?xml version="1.0" encoding="UTF-8"?>
<edmx:Edmx xmlns:edmx=http://docs.oasis-open.org/odata/ns/edmx Version="4.0">
    <edmx:DataServices>
        <Schema xmlns=http://docs.oasis-open.org/odata/ns/edm Namespace="Flows">
            <EntityType Name="Flow0">
                <Key>
                    <PropertyRef Name="Id"/>
                </Key>
                <Property Name="Id" Type="Edm.Int32" Nullable="false"/>
                <Property Name="Continent" Type="Edm.String">
                    <Annotation Term="sap.label">
                        <String>Continent</String>
                    </Annotation>
                    <Annotation Term="sap.objectKey">
                        <String>DP0.DO4</String>
                    </Annotation>
                    <Annotation Term="sap.qualification">
                        <String>DIMENSION</String>
                    </Annotation>
```

```
                </Property>
                <Property Name="Country" Type="Edm.String">
                    <Annotation Term="sap.label">
                        <String>Country</String>
                    </Annotation>
                    <Annotation Term="sap.objectKey">
                        <String>DP0.DO6</String>
                    </Annotation>
                    <Annotation Term="sap.qualification">
                        <String>DIMENSION</String>
                    </Annotation>
                </Property>
                <Property Name="NumberofOrders" Type="Edm.Double">
                    <Annotation Term="sap.label">
                        <String>Number of Orders</String>
                    </Annotation>
                    <Annotation Term="sap.objectKey">
                        <String>DP0.DO43</String>
                    </Annotation>
                    <Annotation Term="sap.qualification">
                        <String>MEASURE</String>
                    </Annotation>
                    <Annotation Term="sap.projectionFunction">
                        <String>Sum</String>
                    </Annotation>
                </Property>
                <Property Name="LateOrders" Type="Edm.Double">
                    <Annotation Term="sap.label">
                        <String>Late Orders</String>
                    </Annotation>
                    <Annotation Term="sap.objectKey">
                        <String>DP0.DO45</String>
                    </Annotation>
                    <Annotation Term="sap.qualification">
                        <String>MEASURE</String>
                    </Annotation>
                    <Annotation Term="sap.projectionFunction">
                        <String>Sum</String>
                    </Annotation>
                </Property>
                <Property Name="LateOrders_1" Type="Edm.Double">
                    <Annotation Term="sap.label">
                        <String>% Late Orders</String>
                    </Annotation>
                    <Annotation Term="sap.objectKey">
                        <String>DP0.DO46</String>
                    </Annotation>
                    <Annotation Term="sap.qualification">
                        <String>MEASURE</String>
                    </Annotation>
                    <Annotation Term="sap.projectionFunction">
                        <String>Sum</String>
                    </Annotation>
                </Property>
            </EntityType>
            <EntityContainer Name="Flows">
                <EntitySet Name="Flows0" EntityType="Flows.Flow0"/>
            </EntityContainer>
            <Annotation Term="sap.isPartial">
                <Bool>false</Bool>
            </Annotation>
        </Schema>
    </edmx:DataServices>
</edmx:Edmx>
```

Related Information

Getting the List of Queries [page 181]

7.4.3 Getting the OData Flow Content

Usage

Returns the data of the specified flow.

> **Note**
>
> The OData service cannot return data page by page.

Request

```
GET /queries/<queryID>/data.svc/<flowName>
```

Response

Response type: `application/xml` or `application/json`

The data of a flow is represented by an Atom feed or an array of JSON objects.

Aside from a series of elements generated automatically (`title`, `updated`, `author`, `link`, and so on), the `<feed>` has the following children:

- `<id>`, which contains the request URL for the specified flow
- A series of `<entry>`, which represents a row of the flow

Aside from generated elements, an `<entry>` has the following children:

- `<id>`, which contains the request URL for a row specified by `<rowIndex>` (`/queries/<queryID>/data.svc/<flowName>(<rowIndex>)`). Row indexes are generated automatically.
- `<content>`, which contains the data.

The data appears in the `<m:properties>` tag. This tag has as many children as the query contains metadata. The tag name of each child is `<d:name>`, where `name` is a label returned by the OData flow metadata. See Getting the OData Flow Metadata [page 207].

Specifying the Request

You can add several parameters to the request to get particular results. The following table describes the possible requests and the type and meaning of the corresponding responses. These parameters are optional. See examples in the next sections.

Request	Response Description	Response Type
/queries/<queryID>/data.svc/<flowName>/$count	The count of records for the specified query flow.	application/text
/queries/<queryID>/data.svc/<flowName>(<rowIndex>)	A row at the <rowIndex> index for the specified query flow. <rowIndex> is an integer.	• application/xml • application/json
/queries/<queryID>/data.svc/<flowName>(<rowIndex>)/<fieldName>	Information about an object specified by <fieldName> from the row at <rowIndex> index in the specified query flow. <fieldName> corresponds to the Property name.	• application/xml • application/json
/queries/<queryID>/data.svc/<flowName>(<rowIndex>)/<fieldName>/$value	The raw value of an object specified by <fieldName> from the row at <rowIndex> in the specified query flow.	application/text
/queries/<queryID>/data.svc/<flowName>?$skip=<offset>	Query results in chunks, starting with the record number specified by the <offset> position in the specified query flow.	• application/xml • application/json
/queries/<queryID>/data.svc/<flowName>?$top=<limit>	The first <limit> records in chunks of the specified query flow.	• application/xml • application/json
/queries/<queryID>/data.svc/<flowName>?$skip=<offset>&$top=<limit>	The first <limit> records in chunks, starting with the record number specified by the <offset> position in the specified query flow.	• application/xml • application/json

Related Information

Getting the List of Queries [page 181]
OData Content - Getting All Data [page 211]
OData Content - Getting the Row Count [page 214]
OData Content - Getting the First Row [page 215]
OData Content - Getting Property Content [page 215]
OData Content - Getting Property Raw Content [page 216]
OData Content - Getting Content After Offset [page 216]

OData Content - Getting the First N Rows [page 219]
OData Content - Getting the First N Rows After Offset [page 220]

7.4.4 OData Content - Getting All Data

Example

The following request returns all the data that the `Flows0` flow contains.

`GET sl/v1/queries/5924817691768289990/data.svc/Flows0`

Response type: `application/xml`

```
<?xml version="1.0" encoding="UTF-8"?>
<a:feed xmlns:a=http://www.w3.org/2005/Atom xmlns:m=http://docs.oasis-
open.org/odata/ns/metadata xmlns:d=http://docs.oasis-open.org/odata/ns/data
m:context="$metadata#Flows0">
    <a:id>http://localhost:9080/biprws/sl/v1/queries/5924817691768289990/Flows0</
a:id>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:10:32Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">0</d:Id>
                <d:Continent>North America</d:Continent>
                <d:Country>USA</d:Country>
                <d:NumberofOrders m:type="Double">1468.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">88.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0599455040871935</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:10:32Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">1</d:Id>
                <d:Continent>North America</d:Continent>
                <d:Country>Canada</d:Country>
                <d:NumberofOrders m:type="Double">713.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">44.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.061711079943899</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
```

```
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:10:32Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">2</d:Id>
                <d:Continent>South America</d:Continent>
                <d:Country>Brasil</d:Country>
                <d:NumberofOrders m:type="Double">629.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">34.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0540540540540541</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:10:32Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">3</d:Id>
                <d:Continent>Asia</d:Continent>
                <d:Country>Korea</d:Country>
                <d:NumberofOrders m:type="Double">251.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">15.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0597609561752988</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:10:32Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">4</d:Id>
                <d:Continent>South America</d:Continent>
                <d:Country>Mexico</d:Country>
                <d:NumberofOrders m:type="Double">260.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">15.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0576923076923077</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:10:32Z</a:updated>
        <a:author>
```

```
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">5</d:Id>
                <d:Continent>Asia</d:Continent>
                <d:Country>Singapore</d:Country>
                <d:NumberofOrders m:type="Double">204.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">12.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0588235294117647</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:10:32Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">6</d:Id>
                <d:Continent>Asia</d:Continent>
                <d:Country>Japan</d:Country>
                <d:NumberofOrders m:type="Double">820.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">42.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.051219512195122</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:10:32Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">7</d:Id>
                <d:Continent>Asia</d:Continent>
                <d:Country>Thailand</d:Country>
                <d:NumberofOrders m:type="Double">192.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">7.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0364583333333333</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:10:32Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
```

```
            <m:properties>
                <d:Id m:type="Int32">8</d:Id>
                <d:Continent>South America</d:Continent>
                <d:Country>Argentina</d:Country>
                <d:NumberofOrders m:type="Double">203.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">11.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0541871921182266</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
</a:feed>
```

Response type: `application/json`

```
{"d":[
    {"__metadata":
        {"uri": "http://w2k8x64sp2:6405/biprws/sl/v1/queries/6089913651317040730/
data.svc/Flow0(0)",
         "type": "Flows._Flow0"
       },
     "Id": 0,
     "City": "Chicago",
     "Customer": "Steve Rob",
    },
    {"__metadata":
        {"uri": "http://w2k8x64sp2:6405/biprws/sl/v1/queries/6089913651317040730/
data.svc/Flow0(1)",
         "type": "Flows._Flow0"
        },
     "Id": 1,
     "City": "Chicago",
     "Customer": "eMarket",
    },
    {
     ...
    },
     ...
    ]
}
```

7.4.5 OData Content - Getting the Row Count

Example

The following request returns the numbers of rows in the specified flow. This is also the number of `<entry>` elements in the feed.

`GET /queries/6089913651317040730/data.svc/Flow0/$count`

Response type: `application/text`

```
120
```

7.4.6 OData Content - Getting the First Row

Example

The following request returns the data of the first row of the `Flows0` flow.

Request:

`GET sl/v1/queries/5045279711854322556/data.svc/Flows0(0)`

Response type: `application/xml`

```
<?xml version="1.0" encoding="UTF-8"?>
<a:entry xmlns:a=http://www.w3.org/2005/Atom xmlns:m=http://docs.oasis-
open.org/odata/ns/metadata xmlns:d=http://docs.oasis-open.org/odata/ns/data
m:context="$metadata#Flows0">
    <a:title/>
    <a:summary/>
    <a:updated>2022-10-19T17:36:41Z</a:updated>
    <a:author>
        <a:name/>
    </a:author>
    <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
    <a:content type="application/xml">
        <m:properties>
            <d:Id m:type="Int32">0</d:Id>
            <d:Continent>North America</d:Continent>
            <d:Country>USA</d:Country>
            <d:NumberofOrders m:type="Double">1468.0</d:NumberofOrders>
            <d:LateOrders m:type="Double">88.0</d:LateOrders>
            <d:LateOrders_1 m:type="Double">0.0599455040871935</d:LateOrders_1>
        </m:properties>
    </a:content>
</a:entry>
```

Response type: `application/json`

```
{"d":[
    {"__metadata":
        {"uri": "http://w2k8x64sp2:6405/biprws/sl/v1/queries/6089913651317040730/
data.svc/Flows0(0)",
         "type": "Flows._Flow0"
       },
     "Id": 0,
     "City": "Chicago",
     "Customer": "Steve &amp; Rob",
    }
]}
```

7.4.7 OData Content - Getting Property Content

Example

The following request returns the value of the `Customer` property on the specified row of the `Flows0` flow.

`GET sl/v1/queries/5529935817033291697/data.svc/Flows0(0)/Continent`

Response type: `application/xml`

```
<?xml version="1.0" encoding="UTF-8"?>
<m:value xmlns:m=http://docs.oasis-open.org/odata/ns/metadata m:context="../
$metadata#Flows0/Continent">North America</m:value>
```

Response type: `application/json`

```
{"d": [
    {"__metadata":
        {"uri": "http://w2k8x64sp2:6405/biprws/sl/v1/queries/6089913651317040730/
data.svc/Flow0(3)",
         "type": "Flows._Flow0"
       },
     "Customer": "Chicago Mall",
    },
]}
```

7.4.8 OData Content - Getting Property Raw Content

Example

The following request returns the value of the `Customer` property for the specified row of the `Flows0` flow.

`GET /queries/6089913651317040730/data.svc/Flows0(3)/Customer/$value`

Response type: `application/text`

```
Chicago Mall
```

7.4.9 OData Content - Getting Content After Offset

Example

The following request returns the content of the `Flows0` flow starting with the third row.

`GET sl/v1/queries/5689774110331453308/data.svc/Flows0?$skip=2`

Response type: `application/xml`

```
<?xml version="1.0" encoding="UTF-8"?>
<a:feed xmlns:a=http://www.w3.org/2005/Atom xmlns:m=http://docs.oasis-
open.org/odata/ns/metadata xmlns:d=http://docs.oasis-open.org/odata/ns/data
m:context="$metadata#Flows0">
    <a:id>http://localhost:9080/biprws/sl/v1/queries/5689774110331453308/Flows0</
a:id>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:49:01Z</a:updated>
        <a:author>
            <a:name/>
```

```
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">2</d:Id>
                <d:Continent>South America</d:Continent>
                <d:Country>Brasil</d:Country>
                <d:NumberofOrders m:type="Double">629.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">34.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0540540540540541</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:49:01Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">3</d:Id>
                <d:Continent>Asia</d:Continent>
                <d:Country>Korea</d:Country>
                <d:NumberofOrders m:type="Double">251.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">15.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0597609561752988</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:49:01Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">4</d:Id>
                <d:Continent>South America</d:Continent>
                <d:Country>Mexico</d:Country>
                <d:NumberofOrders m:type="Double">260.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">15.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0576923076923077</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:49:01Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
```

```
                <d:Id m:type="Int32">5</d:Id>
                <d:Continent>Asia</d:Continent>
                <d:Country>Singapore</d:Country>
                <d:NumberofOrders m:type="Double">204.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">12.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0588235294117647</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:49:01Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">6</d:Id>
                <d:Continent>Asia</d:Continent>
                <d:Country>Japan</d:Country>
                <d:NumberofOrders m:type="Double">820.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">42.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.051219512195122</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:49:01Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">7</d:Id>
                <d:Continent>Asia</d:Continent>
                <d:Country>Thailand</d:Country>
                <d:NumberofOrders m:type="Double">192.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">7.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0364583333333333</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:49:01Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">8</d:Id>
                <d:Continent>South America</d:Continent>
                <d:Country>Argentina</d:Country>
                <d:NumberofOrders m:type="Double">203.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">11.0</d:LateOrders>
```

```
                <d:LateOrders_1 m:type="Double">0.0541871921182266</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
</a:feed>
```

7.4.10 OData Content - Getting the First N Rows

Example

The following request returns the content of the first four rows of the `Flows0`.

`GET sl/v1/queries/7332600476947377083/data.svc/Flows0?$top=4`

Response type: `application/xml`

```
<?xml version="1.0" encoding="UTF-8"?>
<a:feed xmlns:a=http://www.w3.org/2005/Atom xmlns:m=http://docs.oasis-
open.org/odata/ns/metadata xmlns:d=http://docs.oasis-open.org/odata/ns/data
m:context="$metadata#Flows0">
    <a:id>http://localhost:9080/biprws/sl/v1/queries/7332600476947377083/Flows0</
a:id>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:51:36Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">0</d:Id>
                <d:Continent>North America</d:Continent>
                <d:Country>USA</d:Country>
                <d:NumberofOrders m:type="Double">1468.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">88.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0599455040871935</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:51:36Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">1</d:Id>
                <d:Continent>North America</d:Continent>
                <d:Country>Canada</d:Country>
                <d:NumberofOrders m:type="Double">713.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">44.0</d:LateOrders>
```

```
                <d:LateOrders_1 m:type="Double">0.061711079943899</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:51:36Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">2</d:Id>
                <d:Continent>South America</d:Continent>
                <d:Country>Brasil</d:Country>
                <d:NumberofOrders m:type="Double">629.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">34.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0540540540540541</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:51:36Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">3</d:Id>
                <d:Continent>Asia</d:Continent>
                <d:Country>Korea</d:Country>
                <d:NumberofOrders m:type="Double">251.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">15.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0597609561752988</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
</a:feed>
```


7.4.11 OData Content - Getting the First N Rows After Offset

Example

The following request returns the content of the first ten rows of the `Flows0`, starting with the second row.

`GET sl/v1/queries/5634093560637853592/data.svc/Flows0?$skip=2&$top=3`

Response type: `application/xml`


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

```
<a:feed xmlns:a=http://www.w3.org/2005/Atom xmlns:m=http://docs.oasis-
open.org/odata/ns/metadata xmlns:d=http://docs.oasis-open.org/odata/ns/data
m:context="$metadata#Flows0">
    <a:id>http://localhost:9080/biprws/sl/v1/queries/5634093560637853592/Flows0</
a:id>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:53:08Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">2</d:Id>
                <d:Continent>South America</d:Continent>
                <d:Country>Brasil</d:Country>
                <d:NumberofOrders m:type="Double">629.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">34.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0540540540540541</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:53:08Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">3</d:Id>
                <d:Continent>Asia</d:Continent>
                <d:Country>Korea</d:Country>
                <d:NumberofOrders m:type="Double">251.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">15.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0597609561752988</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
    <a:entry>
        <a:title/>
        <a:summary/>
        <a:updated>2022-10-19T17:53:08Z</a:updated>
        <a:author>
            <a:name/>
        </a:author>
        <a:category scheme=http://docs.oasis-open.org/odata/ns/scheme
term="#Flows.Flow0"/>
        <a:content type="application/xml">
            <m:properties>
                <d:Id m:type="Int32">4</d:Id>
                <d:Continent>South America</d:Continent>
                <d:Country>Mexico</d:Country>
                <d:NumberofOrders m:type="Double">260.0</d:NumberofOrders>
                <d:LateOrders m:type="Double">15.0</d:LateOrders>
                <d:LateOrders_1 m:type="Double">0.0576923076923077</
d:LateOrders_1>
            </m:properties>
        </a:content>
    </a:entry>
```

```
</a:feed>
```

8 Web Intelligence REST API Reference

Reference of the Web Intelligence REST APIs.

Each API reference provides the following information:

- What the API does
- URL of the HTTP request
- Description of the request query parameters
- Request body if needed (XML or JSON example used)
- Response type, which determines the value of the `accept` header attribute
- Response body (not described if the response is a success or error message)
- Response examples

Managing Documents [page 223]

Managing Reports [page 339]

Drilling on Report Data [page 376]

Managing Report Elements [page 390]

Working With Input Controls [page 499]

Managing Data Providers [page 554]

Managing Personal Data Providers [page 600]

Managing SAP BW Connections and BEx Queries [page 607]

Managing Connections for Free-Hand SQL Data Providers [page 616]

Refreshing Documents [page 619]

Refreshing Data Providers [page 664]

Scheduling Documents [page 667]

Managing Publications [page 677]
A publication is a collection of documents that you send to a mass audience.

Managing Shared Elements [page 679]

Searching for Resources [page 691]

Managing Universes with the Web Intelligence RESTful Web Service SDK [page 695]

Getting Document Content Through OData [page 702]
The OData APIs allows you to explore a Web Intelligence document's content and retrieve its dataset.

8.1 Managing Documents

The Web Intelligence RESTful Web Service SDK provides a series of APIs to work with Web Intelligence documents.

> **⚠ Restriction**
>
> Management of auto-save and auto-recovery configuration is currently not supported.

Creating a Document [page 224]

Getting the List of Documents [page 226]

Getting the Details of a Document [page 228]

Getting a Document's Security Rights for the Current User [page 231]

Saving a Document [page 233]

Copying a Document [page 236]

Purging a Document [page 237]

Working with ZIP files [page 238]

Exporting Documents [page 242]

Updating the State of a Document [page 247]

Deleting a Document [page 250]

Working with Properties [page 251]

Managing Styles, Formats, Fonts, Skins, and Charsets [page 254]

Working with Color Palettes [page 276]

Managing Functions, Operators, and Variables [page 284]

Managing Links [page 295]

Managing Alerters and Tracker Settings [page 301]

Managing Attachments [page 318]

Working with Snapshots [page 321]

Working with Occurrences [page 325]

Working with Referenced Cells [page 331]

A reference is a variable whose definition and content are based on another cell. It is useful whenever you want to leverage data of a cell that has been obtained using a complex formula.

Adding a Cache Entry to a Document [page 337]

8.1.1 Creating a Document

Usage

Creates an empty Web Intelligence document.

Request

`POST /documents`

Request type: `application/xml` or `application/json`

Request body:

```
<document>
    <name>
    <folderId>
```

Element	Type or Value	Description
`<name>`	*string*	The name of the document to be created.
`<folderId>`	*integer*	The BI launch pad folder where the document is created. You can retrieve the folder ID from a GET call to `http://<server-name>:6405/biprws/infostore`.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: XML Example

`POST /documents`

Request body:

```
<document>
    <name>My Document</name>
    <folderId>5151</folderId>
</document>
```

Response body:

```
<success>
   <message>The resource of type "Document" with identifier "5022" has been
successfully created.</message>
   <id>5022</id>
</success>
```

An identifier has been assigned automatically.

Example: JSON Example

`POST /documents`

Request body:

```
{"document":
    {"name": "my document",
     "folderId:"5151"}
}
```

Response body:

```
{"success":
   {"message": "The resource of type "Document" with identifier "5022" has been
successfully created.",
    "id": 5022}
}
```

An identifier has been assigned automatically.

8.1.2 Getting the List of Documents

Usage

Gets the list of Web Intelligence documents stored in the CMS repository.

The documents are sorted by name. The list returned depends on user access rights. You can also specify the number of documents to return and the first document to be used as the start document in the list.

> **Note**
>
> You can also search for documents using the `/searches` API.

Request

`GET: /documents?offset=<offset>&limit=<limit>`

Where:

- `<offset>` indicates the position in the list, from which documents are returned. It must be greater than or equal to 0. The default value is 0. This parameter is optional.
- `<limit>` indicates the number of documents in the list. The range is [0, 50]. The default value is 10. This parameter is optional.

Response

Response type: `application/xml` or `application/json`

Response body: the list of Web Intelligence documents sorted by name and identified by the following elements:

Element	Type or Value	Description
`<id>`	*integer*	The document ID
`<cuid>`	*string*	The unique document ID
`<name>`	*string*	The document name
`<description>`	*string*	The optional document description
`<folderId>`	*integer*	The folder ID

Example: To Get Six Documents from the CMS (XML)

`GET /documents?limit=6`

Response body:

```
<documents>
    <document>
        <id>6804</id>
        <cuid>AbrVz6E95lNMtjZk4KLzPuM</cuid>
        <name>BOF-ROLE_AA_ActiveHierOnCtry - Levels 02 and 03</name>
        <folderId>7611</folderId>
    </document>
    <document>
        <id>6861</id>
        <cuid>AUwfvPdEG3xOoLNoNEzDquQ</cuid>
        <name>BOF-QRY_SIMPLE - Filter not in Result</name>
        <description>This is a sample for BOF-QRY_SIMPLE - Filter not in Result</
description>
        <folderId>7611</folderId>
    </document>
    <document>
        <id>6869</id>
        <cuid>Ac1wDFEOuvNHtovI4H.anSM</cuid>
        <name>BOF-QRY_SIMPLE - Filter on Country</name>
        <folderId>48162</folderId>
    </document>
    <document>
        <id>7048</id>
        <cuid>AUiVpegCKnFBh4AX8YxMT5A</cuid>
        <name>BOF-QRY_SIMPLE - Filter on Prompt not in Result</name>
        <folderId>7611</folderId>
    </document>
    <document>
        <id>7125</id>
        <cuid>AWa48YHHT6FFiUMP.T5wtBs</cuid>
        <name>BOF-QRY_SIMPLE - Query Prompt not in Result</name>
        <folderId>7611</folderId>
    </document>
    <document>
        <id>5121</id>
```

```
        <cuid>ARnGKBoLv7lDuK_UWMUli5Q</cuid>
        <name>Chart demo</name>
        <description>Shows how data can be visualized on different kinds of
charts.
            And the features supported on the charts. </description>
        <folderId>5127</folderId>
    </document>
</documents>
```

Example: To Get Three Documents from the CMS (JSON)

`GET /documents?limit=3`

Response body:

```
{"documents":
    {"document":[
        {"id":5152,
         "cuid":"AZx1nlTlCMdCvyJ6bWUTC5I",
         "name":"Chart Demo",
         "description":"Shows how data can be visualized on different kinds of
charts. And the features supported on the charts.",
         "folderId":5151,
        {"id":5169,
         "cuid":"AW4AVT1AUhVAogA6P7OQv9c",
         "name":"Charting Samples",
         "folderId":5150,
         "state":"Unused"},
        {"id":5157,
         "cuid":"AS9ukIRdciZLuUS6ESGVRBg",
         "name":"Drill Demo",
         "description":"This document demonstrates the drill functionality on
tables and Charts",
         "folderId":5151}]
    }
}
```

Related Information

Searching for Resources [page 691]

8.1.3 Getting the Details of a Document

Usage

Gets the details of a Web Intelligence document.

Request

`GET /documents/<documentID>?trackerDocumentId=<trackerDocumentID>`

Where:

- `<trackerDocumentID>`: identifier of a reference document for trackdata feature. Optional. Must be provided only when the document state is unused.

Response

Response type: `application/xml` or `application/json`

Response body: the details of the document identified by the following elements:

Element	Type or Value	Description
`<id>`	Integer	The document ID
`<cuid>`	String	The unique document ID
`<name>`	String	The document name
`<description>`	String	The document description
`<folderId>`	Integer	The identifier of the folder of the CMS repository that contains the document
`<path>`	String	The path to the document in the CMS repository directory
`<updated>`	DateTime	The date and the time of the last update
`<scheduled>`	Boolean	`true` if the document has been scheduled
`<state>`	`Unused`| `Original`| `Modified`	The state of document used by the service. Possibles values are: • `Unused`, if the document has not been loaded in the web service container • `Original`, if the document has been loaded by the web service but not modified • `Modified`, if the user sent requests which altered the state of the document
`<createdBy>`	String	The name of the document creator
`<lastAuthor>`	String	The name of the last person who modified the document
`<size>`	Integer	The size of the document in bytes
`<refreshOnOpen>`	Boolean	`true` if the document is set to be refreshed at the open time

Example: Getting the Details of a Document (XML)

```
GET /documents/4958
```

Response body:

```
<document>
    <id>4958</id>
    <cuid>AQtkbbSqN4NOj3ydf.Sw1lY</cuid>
    <name>Formatting Sample</name>
    <description>This is a sample document showing formatting capabilities.</
description>
    <folderId>8246</folderId>
    <path>My Folders/web Intelligence</path>
    <updated>2012-09-05T14:25:05.277+02:00</updated>
    <scheduled>true</scheduled>
    <state>Unused</state>
    <createdBy>Administrator</createdBy>
    <lastAuthor>User1</lastAuthor>
    <size>29410</size>
    <refreshOnOpen>false</refreshOnOpen>
</document>
```

Example: Getting the Details of a Document (JSON)

```
GET /documents/5152
```

```
{"document":
    {"id":5152,
     "cuid":"AZx1nlTlCMdCvyJ6bWUTC5I",
     "name":"Chart Demo",
     "description":"Shows how data can be visualized on different kinds of
charts. And the features supported on the charts.",
     "folderId":5151,
     "path":"Public Folders\/Web Intelligence Samples\/Mobile Samples",
     "updated":"2015-04-21T18:10:29.052+02:00",
     "scheduled":false,
     "state":"Unused",
     "createdBy":"Administrator",
     "lastAuthor":"Administrator",
     "size":298874,
     "refreshOnOpen":false}
}
```

Related Information

Getting the List of Documents [page 226]

8.1.4 Getting a Document's Security Rights for the Current User

Usage

This URL returns the list of available rights of the document.

Request

```
GET /documents/<documentID>/rights
```

Response

Response type: `application/xml` or `application/json`

The available rights are:

- `add_objects_to_folder`
- `allow_personal_data_providers`
- `app_allow_autosave`
- `app_create_edit_input_controls`
- `app_edit_document_formatting`
- `app_edit_my_preferences`
- `app_enable_context_menus`
- `app_hide_show_toolbox`
- `app_lp_document_filters`
- `app_lp_document_summary`
- `app_publish_document_content`
- `app_re_create_edit_alerter`
- `app_re_create_edit_break`
- `app_re_create_edit_predefined_calculation`
- `app_re_create_edit_report_filters`
- `app_re_create_edit_sort`
- `app_re_insert_element`
- `app_view_sql`
- `change_TDC_format`
- `create_documents`
- `delete_objects`
- `disable_automatic_refresh_on_open`
- `disable_export_to_BIOD`

- disable_import_from_BIOD
- edit_documents
- edit_objects
- edit_query_sql
- enable_shared_objects
- insert_shared_objects
- merge_for_synchro
- publish_documents_real
- rc_enable_use
- rc_export
- rc_import
- rc_local_save
- rc_print
- rc_remove_document_security
- rc_save_for_all_users
- rc_send_by_mail
- read_corporate_documents
- use_TDC
- use_formula_language
- use_interactive_viewing

Example: XML Format

GET /documents/6702/rights

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<rights>
    <right>
        <id>refresh_documents</id>
    </right>
    <right>
        <id>use_list_of_values</id>
    </right>
    <right>
        <id>refresh_list_of_values</id>
    </right>
    ...
</rights>
```

Example: JSON Format

GET /documents/6702/rights

```
{
    "rights": {
        "right": [{
```

```
            "id": "refresh_documents"
        },
        {
            "id": "use_list_of_values"
        },
        ...
        }]
    }
}
```

8.1.5 Saving a Document

Usage

Saves a Web Intelligence document in the CMS repository.

When you save a document, it is copied to the destination folder and an identifier is assigned automatically. This action creates a new version of the document. If the document was open by the REST Web Service, it is replaced by the new created one.

Request

```
POST /documents/<documentID>?overwrite=<boolean>&withComments=<boolean>
```

```
POST /documents/<documentID>/occurences/<occurenceId>?
overwrite=<boolean>&withComments=<boolean>
```

Where:

- `overwrite` (optional) enables to avoid overwriting an existing version of the document. By default, the value is set to `true`. If set to `false`, an error will be returned if the document already exists.
- `withComments` (optional) enables to copy all the comments in the source document to the saved document. The default value is `false`. If set to `true`, all the comments of the source document will be copied to the target document.

Request type: `application/xml` or `application/json`

Request body:

```
<document>
    <name>
    <description>
    <keywords>
    <folderId>
    <categories>
        <category>
            <id>
    <properties>
        <property>
```

Element	Type or Value	Description
`<name>`	String	The name of the document to save.
`<description>`	String	Optional. The description of the document to save.
`<keywords>`	String	Optional.Keywords to tag the document to save.
`<folderId>`	Integer	The identifier of the BI launch pad destination folder. **Note** Use `-1` as folder identifier to save the document to the same folder.
`<categories>`	Integer	Optional. The identifier of the category (public or personal) assigned to the document
`<properties>`	Boolean	Optional. Name of the properties you can assign to the new document, namely *Refresh on open* and *Regional Formatting*. **Note** If you don't specify the `refreshonopen` and `permanentregionalformatting` properties, they inherit the document's value.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`POST /documents/1234`

Request body:

```
<document>
    <name>doc_tables_V2</name>
    <description>2nd version</description>
    <keywords>webi tables</keywords>
    <categories>
        <category>
            <id>6598</id>
    </category>
    <category>
          <id>6594</id>
    </category>
    </categories>
    <properties>
        <property key="refreshonopen">true</property>
        <property key="permanentregionalformatting">true</property>
  </properties>
```

```
</document>
```

Response:

```
<success>
    <message>The resource of type "Document" with identifier "9217" has been
successfully created.</message>
    <id>9217</id>
    <details>
        <property key="occurrenceId">0</property>
    <details>
</success>
```

An identifier has been assigned automatically.

Example:

Saving a document with comments, and avoiding overwriting if a local version already exists (an error message will be returned).

`POST /documents/7400?overwrite=false&withComments=true`

Request body:

```
<document>
    <name>Document Save As Example</name>
    <keywords>Save as</keywords>
    <folderId>6773</folderId>
    <categories>
        <category>
            <id>5980</id>
        </category>
        <category>
            <id>6571</id>
        </category>
    <categories>
    <properties>
       <property key="refreshonopen">true</property>
       <property key="permanentregionalformatting">true</property>
</document>
```

Response:

```
<success>
    <message>The resource of type "Document" with identifier "32271" has been
successfully created.</message>
    <id>32271</id>
</success>
```

> **Note**
>
> - Overwriting a document having comments with the option "`withComments=false`" will not remove its comments
> - When saving an occurence of a document as a document, this occurence remains opened. It's up to the user to close/keep it.

8.1.6 Copying a Document

Usage

Copies a Web Intelligence document referenced by its ID. Unlike the Save request, this does not open the document in memory.

Compared to the Saving a Document call, this copy doesn't open the document in memory.

Request

`POST /documents?sourceId=<documentID>`

Request type: `application/xml` or `application/json`

Request body (optional):

```
<document>
    <name>
    <folderId>
```

Element	Type or Value	Description
`<name>`	*string*	The name of the copied document in the BI launch pad folder. If not specified, the service assigns a name to the document automatically.
`<folderId>`	*integer*	The BI launch pad folder where the document is copied. If not specified, the folder is the one of the original document.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: Copying a Document (XML)

`POST /documents?sourceId=4990`

Request body:

```
<document>
    <name>Copy of 4990</name>
</document>
```

Response:

```
<success>
    <message>The resource of type "document" with identifier "5875" has been
successfully created.</message>
    <id>5875</id>
</success>
```

The name "Copy of 4990" and ID "5875" have been assigned to the copied document. The folder is the same as the original one.

Example: Copying a Document (JSON)

```
POST /documents?sourceId=5152
```

The original document is named "Chart Demo".

Response:

```
{"success":
    {"message":"The resource of type \"Document\" with identifier \"6585\" has
been successfully created.",
     "id":6585}
}
```

The name "Chart Demo[1]" and ID "6585" have been assigned to the copied document. The folder is the same as the original one.

Related Information

Getting the List of Documents [page 226]

8.1.7 Purging a Document

Usage

Purges a document or its data providers.

Request

```
PUT /documents/<documentID>?
purge=<boolean>&purgeOptions=<string>&dataProviderIds=<dataProviderID>
```

Where:

- `purge` is an optional boolean that specifies whether the application should purge the document. It's set to false by default.
- `purgeOptions` is an optional string parameter that manages the purge. For now, `prompt` is the only value allowed. If you don't set this parameter or leave it empty, the application only purges the data provider.
- `dataProviderID` is an optional string that specifies the data providers whose parameters should be retrieved. If you have multiple data providers, make sure to separate them using a comma.

Request type: `application/xml` or `application/json`

Example

```
PUT /documents/11149?purge=true&purgeOptions=prompts&dataproviderIds=DP0,DP1
```

The response is a message stating the success or failure of the request.

Reponse body:

```
<success>
    <message>The resource of type "Data providers" with identifier "[DP0, DP1]"
has been successfully updated.</message>
    <id>[DP0, DP1]</id>
</success>
```

8.1.8 Working with ZIP files

You can archive the reports of a Web Intelligence document as HTML files into a ZIP file.

Creating a ZIP File from a Document [page 238]

Getting a Document as a ZIP file [page 240]

Deleting a ZIP File [page 241]

8.1.8.1 Creating a ZIP File from a Document

Usage

Creates a ZIP file archiving the document reports of your choice as HTML files.

The file is created on the server.

> ⚠ Caution
>
> - You get the ZIP file by calling `GET /documents/<documentID>/zips/<zipID>?fileName=<fileName>`
> - The ZIP file is deleted after the user session closes.

Request

`POST /documents/<documentID>/zips?reportIds=<reportID[,...]>`

Where:

- `<reportID[,...]>` is the list of the report identifiers to be archived in the ZIP file, separated with a comma.
  This parameter is optional. If not present, all the reports of the document are archived.

Request body: the script that allows the file archiving. See example below.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`POST /documents/14695/zips?reportIds=1,3`

Request body:

```
<zipfile>
    <zipentry name="index.html">
        <![CDATA[
            <html>
                <head>
                    <title>Save As HTML (customized)</title>
                    <script language="javascript" src="reports.js"></script>
                </head>
                <body bgcolor="#EEEEEE">
                    <div id='content'></div>
                    <script>
                    var content = document.getElementById('content');
                    content.innerHTML = '';
                    for (var key in reportIds) {
                    var reportLink = "<a href='" + key + "/report.html'>";
reportLink += reportIds[key]['name'];
                    reportLink += '</a>'; reportLink += '<br/>';
content.innerHTML += reportLink; }
                    </script>
                </body>
```

```
            </html>
        ]]>
    </zipentry>
    <zipentry name="report.html">
        <![CDATA[
            <html>
                <head></head>
                <body></body>
            </html>
        ]]>
    </zipentry>
</zipfile>
```

Response:

```
<success>
    <message>The resource of type "ZipFile" with identifier "0" has been
successfully created.</message>
    <id>0</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting a Document as a ZIP file [page 240]

8.1.8.2 Getting a Document as a ZIP file

Usage

Returns a ZIP file that archives the document reports of your choice as HTML files.

> **Note**
>
> - You create the ZIP file by calling `POST /documents/<documentID>/zips?reportIds=<reportID[,...]>`.
> - You can export a document with all of its reports as a ZIP file by calling `GET /documents/<documentID>?fileName=<fileName>`.
> - Exports to HTML will be optimized for the end-user browser if you use the User-Agent HTTP header in the REST API call.

Request

`GET /documents/<documentID>/zips/<zipID>?fileName=<fileName>`

Where:

- `<fileName>` is a string that defines the name of the ZIP file. Maximum string length is 96. This parameter is optional. If set, the HTML output is zipped in a `fileName.zip` file and sent as attachment.

Response

Response type: `application/zip`

Response: the document as a ZIP file.

Example

```
GET /documents/14746/zips/0?fileName=myZipFile
```

Related Information

Getting the List of Documents [page 226]
Exporting a Document in Listing Mode [page 243]

8.1.8.3 Deleting a ZIP File

Usage

Deletes a ZIP file archiving reports of a Web Intelligence document.

Request

```
DELETE /documents/<documentID>/zips/<zipID>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/14746/zips/0
```

Response:

```
<success>
    <message>The resource of type "ZipFile" with identifier "0" has been
successfully removed.</message>
    <id>0</id>
</success>
```

Related Information

Getting the List of Documents [page 226]

8.1.9 Exporting Documents

You can export either a whole Web Intelligence document or some pages of the document.

> **Note**
>
> Since 4.2 SP4, if drill mode and drill output mode are both activated for at least one of the reports, then drill information is included in `text/xml` output.

> **Note**
>
> Before 4.2 SP4, default value for dpi was set to 300 for PDF and 96 for other output types. Since 4.2 SP4, the default behavior is to keep the server's settings.

> **Note**
>
> Before 4.2 SP4, the default value for Excel data optimization was set to false. Since 4.2 SP4, the default behaviour is to keep the server's settings.

Exporting a Document in Listing Mode [page 243]

Exporting a Document as a Series of Pages [page 245]

8.1.9.1 Exporting a Document in Listing Mode

Usage

Exports the whole Web Intelligence document to the desired format:

- XML
- zipped HTML
- PDF
- Microsoft Excel 2003
- Microsoft Excel 2007

> **Note**
>
> Exports to HTML will be optimized for the end-user browser if you use the User-Agent HTTP header in the REST API call.

Request

```
GET /documents/<documentID>?<optional_parameters>
```

Optional Parameters

Parameter	Description	Supported Formats
`dpi`	Resolution in dots per inch (dpi) for generated charts. Value between 75 and 9600. Default is 300 for PDF format, 96 for all other formats.	All
`unit`	A string that defines the unit that sizes will be reported in. Values are `"metric"`, `"millimeter"`, `"point"`, and `"pixel"` (default).	XML
`rawValues`	Boolean. Default is `false`. If `true`, the raw values and their types are exported with the formatted values.	XML
`fileName`	A string that defines the name of the ZIP file. If set, the HTML output is zipped to a `fileName.zip` file. Maximum string length is 96.	ZIP
`optimized`	Boolean. Default is `false`. If `true`, the generated output is optimized for calculations inside Microsoft Excel.	Microsoft Excel 2003 and Microsoft Excel 2007

Response

Response type:

- `text/xml`
- `application/zip` for zipped HTML
- `application/pdf`
- `application/vnd.ms-excel` for Microsoft Excel 2003
- `application/vnd.openxmlformats-officedocument.spreadsheetml.sheet` for Microsoft Excel 2007

The response is the file whose name is the document identifier with the output extension type.

Example: XML

Exports the XML result in the `xmlfile.xml`, with a resolution of 150 dpi for any generated charts.

```
curl -G -s -H "accept:text/xml" -H X-SAP-LogonToken:"""%tokenValue%"""
 "<base_webi_REST_URL>/documents/8022?dpi=150&size=pixel" > xmlfile.xml
```

Example: PDF

Exports the result with the file name `exportpdf.pdf`. The default resolution of 300 dpi for any generated charts is used.

```
curl -G -s -H "accept:application/pdf" -H X-SAP-LogonToken:"""%tokenValue%"""
 <base_webi_REST_URL>/documents/8022" > exportpdf.pdf
```

Example: Microsoft Excel 2003

Exports the result with the file name `excel2003.xls` file. The output is optimized for calculations inside Microsoft Excel, and any generated charts will have a resolution of 150 dpi.

```
curl -G -s -H "accept:application/vnd.ms-excel" -H X-SAP-
LogonToken:"""%tokenValue%"""
 <base_webi_REST_URL>/documents/8022?optimized=true&dpi=150" > excel2003.xls
```

Example: Microsoft Excel 2007

Exports the result in the `excel2007.xls` file. The output is optimized for calculations inside Microsoft Excel.

```
curl -G -s -H "accept:application/vnd.openxmlformats-
officedocument.spreadsheetml.sheet"
 -H X-SAP-LogonToken:"""%tokenValue%""" <base_webi_REST_URL>/documents/8022?
optimized=true" > excel2007.xls
```

Related Information

Getting the List of Documents [page 226]

8.1.9.2 Exporting a Document as a Series of Pages

Usage

Exports a Web Intelligence document as a series of pages in one of the following formats:

- XML
- PDF
- Microsoft Excel 2003
- Microsoft Excel 2007

Request

```
GET /documents/<documentID>/pages?<optional_parameters>
```

Optional Parameters

Parameter	Description	Supported Formats
`dpi`	Resolution in dots per inch (dpi) for generated charts. Value between 75 and 9600. Default is 300 for PDF format, 96 for all other formats.	All
`mode`	`normal` or `quickDisplay`.	All
`orientation`	Page orientation. Use to force a specific page orientation. Values are `portrait` and `landscape`.	If `mode=normal`
`widthScaling`	Number of pages per report displaying in width. The document setting applies by default.	If `mode=normal`
`heightScaling`	Number of pages per report displaying in height. The document setting applies by default.	If `mode=normal`
`unit`	A string that defines the unit that sizes will be reported in. Values are `"metric"`, `"millimeter"`, `"point"`, and `"pixel"` (default).	XML
`rawValues`	Boolean. Default is `false`. If `true`, the raw values and their types are exported with the formatted values.	XML

Parameter	Description	Supported Formats
`optimized`	Boolean. Default is `false`. If `true`, the generated output is optimized for calculations inside Microsoft Excel.	Microsoft Excel 2003 and Microsoft Excel 2007

Response

Response type:

- `text/xml`
- `application/pdf`
- `application/vnd.ms-excel` for Microsoft Excel 2003
- `application/vnd.openxmlformats-officedocument.spreadsheetml.sheet` for Microsoft Excel 2007

The response is a file whose name is the document identifier with the output extension type.

Example: XML

Exports the XML result in `xmlfile.xml`.

```
curl -G -s -H "accept:text/xml" -H X-SAP-LogonToken:"""%tokenValue%"""
 <base_webi_REST_URL>/documents/8022/pages?unit=pixel" > xmlfile.xml
```

Example: PDF

Exports the PDF result in `exportpdffile.pdf`.

```
curl -G -s -H "accept:application/pdf" -H X-SAP-LogonToken:"""%tokenValue%"""
 <base_webi_REST_URL>/documents/8022/pages?mode=quickDisplay" > exportpdffile.pdf
```

Example: Microsoft Excel 2003

Exports the result in `excel2003.xls` file.

```
curl -G -s -H "accept:application/vnd.ms-excel" -H X-SAP-
LogonToken:"""%tokenValue%"""
 <base_webi_REST_URL>/documents/8022/pages?mode=normal" > excel2003.xls
```

Example: Microsoft Excel 2007

Exports the result in `excel2007.xlsx` file.

```
curl -G -s -H "accept:application/vnd.openxmlformats-
officedocument.spreadsheetml.sheet"
 -s -H X-SAP-LogonToken:"""%tokenValue%""" <base_webi_REST_URL>/documents/8022/
pages?optimized=true" > excel2007.xlsx
```

Related Information

Getting the List of Documents [page 226]

8.1.10 Updating the State of a Document

Usage

Changes the state of a Web Intelligence document referenced by its ID.

> → Remember
>
> Use this method to save a document to the CMS repository.

About the State of a Document

The state of a document in the WACS server can be one of the following:

State	Description
`Unused`	An unused document is a document that has not been loaded to the server.
`Original`	An original document is a document that has been loaded to the server, but has not been modified. Its state can be changed to `Unused` to release the available memory, closing the document.
`Modified`	A modified document is a document that has been loaded and modified. Its state can be explicitely changed to `Unused` to dismiss all the current changes. The document is closed to release the available memory. You can also save the document to save these modifications. The state of the document returns to `Original`.

State	Description
`KeepAlive`	Keeps the document open when the user interacts with the document without making any call to the server to prevent a server timeout. **Note** `KeepAlive` is a pseudo-state, and does not change the actual state of the document.

As soon as you open a document, its status becomes `Original`. You can move a document from a `Modified` or `Original` to `Unused` state to discard all document changes and close the document. This releases the memory of the WACS server.

Request

`PUT /documents/<documentID>`

Request type: `application/xml` or `application/json`

Request body:

```
<document>
    <state>
```

Remember

The request body is optional. If `<state>` is present, no other tag is accepted.

Document State Change	Result
From `Original` to `Unused`	The document is not modified and closed.
From `Original` with no body or an empty body	The document is not modified.
From `Modified` to `Unused`	The document is updated and closed.
From `Modified` with no body or an empty body	The document is updated and saved.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: To Close an Unmodified Document

`PUT /documents/8008`

Request body:

```
<document>
    <state>Unused</state>
</document>
```

Response body:

```
<success>
    <message>The resource of type "document" with identifier "8008" has not been
modified.</message>
    <id>8008</id>
</success>
```

Example: To Save a Modified Document

`PUT /documents/9326`

Response body:

```
<success>
    <message>The resource of type "document" with identifier "9326" has been
successfully updated.</message>
    <id>9326</id>
</success>
```

Example: To Keep a Document Alive

`PUT/documents/4575`

Request body:

```
<document>
    <state>KeepAlive</state>
</document>
```

Response body:

The response depends on the state of the document. If the document is in the `Original` or `Unused` state, the message is the following:

```
<success>
     <message>The resource of type "document" with identifier "4575" has not
been modified.</message>
     <id>4575</id>
</success>
```

If the document is in the `Modified` state, then the message is the following:

```
<success>
    <message>The resource of type "document" with identifier "4575" has been
successfully updated.</message>
    <id>4575</id>
</success>
```

If the request fails because the timeout has expired for instance, the call returns the following message:

```
<error>
    <error_code>WSR 00402</error_code>
    <message>An error occurred while keeping the resource of type "document"
with identifier "4575" alive.</message>
</error>
```

Related Information

Getting the List of Documents [page 226]

8.1.11 Deleting a Document

Usage

Deletes a Web Intelligence document referenced by its identifier.

Request

`DELETE /documents/<documentID>`

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`DELETE /documents/8022`

```
<success>
```

```
    <message>The resource of type "Document" with identifier "8022" has been
successfully removed.</message>
    <id>8022</id>
</success>
```

Related Information

Getting the List of Documents [page 226]

8.1.12 Working with Properties

Use these methods to list or edit the document properties that are visible in the "Document Summary" of a document in Web Intelligence.

Getting the Properties of a Document [page 251]

Updating the Properties of a Document [page 253]

8.1.12.1 Getting the Properties of a Document

Usage

Gets the properties of a document referenced by its ID.

Request

```
GET /documents/<documentID>/properties
```

Response

Response type: `application/xml` or `application/json`

Response body: a series of `<property>` elements.

Example

```
GET /documents/7858/properties
```

```
<properties>
    <property key="SI_CUID">ATuas0Mb.mxHpfioi7Xrwfo</property>
    <property key="lastrefreshtime">1352999520</property>
    <property key="lastrefreshduration">1</property>
    <property key="lastsavedby">Administrator</property>
    <property key="enablealternateusertoeditdoc">true</property>
    <property key="creatorversion">14.0.6.894</property>
    <property key="disablequerypanel">false</property>
    <property key="lastrefreshdate">2012, Nov 15 18:12:00 GMT+01:00</property>
    <property key="enhancedViewing">false</property>
    <property key="current_doc_id">9939</property>
    <property key="osversion">?</property>
    <property key="modificationtime">1352999598</property>
    <property key="stripquery">true</property>
    <property key="modificationdate">2012, Nov 15 18:13:18 GMT+01:00</property>
    <property key="name">doc_unv_drill_enable</property>
    <property key="createdby">Administrator</property>
    <property key="tdcactivated">false</property>
    <property key="reportselected">1</property>
    <property key="nameinrepo">doc_unv_drill_enable</property>
    <property key="permanentregionalformatting">false</property>
    <property key="reporterversion">14.0.6.894</property>
    <property key="repositorytype">C</property>
    <property key="locale">en_US</property>
    <property key="refreshonopen">false</property>
    <property key="docrepoid">9939</property>
    <property key="documenttype">WID</property>
    <property key="autosynchro">false</property>
    <property key="hassamplingresults">false</property>
    <property key="tdcmodeauto">true</property>
    <property key="effectiverefreshonopen">false</property>
    <property key="contentlocale">en_US</property>
    <property key="mergeprompts">true</property>
    <property key="documentsize">31070</property>
    <property key="extendmergedimension">false</property>
    <property key="autorefresh">false</property>
    <property key="creationtime">1352999595</property>
    <property key="creationdate">2012, Nov 15 18:13:15 GMT+01:00</property>
    <property key="ispartiallyrefreshed">false</property>
    <property key="nbqaawsconnection">0</property>
    <property key="documentversion">14.0.6.894</property>
</properties>
```

Related Information

Getting the List of Documents [page 226]

8.1.12.2 Updating the Properties of a Document

Usage

Updates the properties of a Web Intelligence document referenced by its identifier.

> ⓘ Note
>
> Certain settings are attributed automatically and cannot be set manually (for example, the last refresh time).

Request

`PUT /documents/<documentID>/properties`

Request body:

```
<properties>
    <property key="string">
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`PUT /documents/9939/properties`

Request body:

```
<properties>
    <property key="refreshonopen">true</property>
</properties>
```

Response:

```
<success>
    <message>The resource of type "properties" has been successfully updated.</
message>
    <id>9939</id>
</success>
```

Related Information

Getting the List of Documents [page 226]

8.1.13 Managing Styles, Formats, Fonts, Skins, and Charsets

The following APIs allow you to manage the visual aspects of your documents and reports.

Getting the Configuration Formats [page 254]

Getting Custom Formats for Numbers [page 256]

Creating Custom Number or Datetime Formats [page 257]

Getting the Font Mappings [page 258]

Getting the Locales [page 259]

Getting the Report Skins [page 260]

Getting the Styles of a Document [page 262]

Getting the Details of a Style [page 263]

Adding a Style to a Document [page 264]

Updating a Style [page 266]

Deleting a Style [page 267]

Getting the CSS of a Document [page 268]

Updating the CSS of a Document [page 269]

Getting the Charsets [page 270]

Getting the Chart Types [page 271]

Getting the Gradient Orientations [page 273]

Aggregating Configuration Calls [page 274]

8.1.13.1 Getting the Configuration Formats

Usage

Gets a list of all declared formats on the Web Intelligence Server for the default locale. Returns the list of default formats and available formats.

The results depends on the Preferred Viewing Locale (PVL). If you need a specific locale setting, the locale must be passed in the header of the HTTP request. See Supporting Multiple Languages [page 55].

Request

```
GET /configuration/formats
```

Response

Response type: `application/xml` or `application/json`

Response body:

```
<format default="Boolean" type="Number|Percent|Currency|DateTime|Boolean|Date|
Time" sample="string">
    <template positive="string"/>
```

Where:

- `default` is `true` if it is a default format
- `type` is the type of any format
- `sample` is a sample value of the corresponding format

Example

```
<formats>
    <format default="true" type="Number" sample="1 234,57">
        <template positive=""/>
    </format>
    <format default="true" type="Currency" sample="1 234,57 €; -1 234,57 €">
        <template positive="#,##0.00' €'"/>
    </format>
    <format default="true" type="DateTime" sample="21/09/2004 20:45:30">
        <template positive="dd'/'MM'/'yyyy HH':'mm':'ss"/>
    </format>
    <format default="true" type="Boolean" sample="vrai; faux">
        <template positive="BOOLEAN"/>
    </format>
    <format default="true" type="Date" sample="21/09/2004">
        <template positive="dd'/'MM'/'yyyy"/>
    </format>
    <format default="true" type="Time" sample="20:45:30">
        <template positive="HH':'mm':'ss"/>
    </format>
    <format type="Number" sample="1 234,57">
        <template positive="STANDARD"/>
    </format>
    <format type="Number" sample="1,234567E3">
        <template positive="SCIENTIFIC"/>
    </format>
    <format type="Number" sample="1235; (1235)">
        <template positive="0" negative="(0)"/>
    </format>
    <format type="Number" sample="1234,57">
         <template positive="0.00"/>
    </format>
    <format type="Number" sample="1 235">
```

```
        <template positive="#,##0"/>
    </format>
    <format type="Currency" sample="1 234,57 €; -1 234,57 €">
        <template positive="#,##0.00' €'"/>
    </format>
    ...
    <format type="Percent" sample="123 456,70%">
        <template positive="#,##0.00[%]'%'"/>
    </format>
</formats>
```

Related Information

Supporting Multiple Languages [page 55]

8.1.13.2 Getting Custom Formats for Numbers

Usage

Gets a list of the custom format numbers defined in a document.

Request

`GET /documents/<documentID>/formats`

Response

Response type: `application/xml` or `application/json`

Response body: description of the custom formats used in the Web Intelligence document.

Example

`GET /documents/4326/formats`

```
<formats>
    <format type="Custom" sample="1,235">
        <template positive="#,##0" negative="[red]#,##0" zero="No value"
undefined="NaN"></template>
    </format>
</formats>
```

In this example, negative numbers are in red.

Related Information

Getting the List of Documents [page 226]

8.1.13.3 Creating Custom Number or Datetime Formats

Usage

Creates a custom format for numbers or datetimes.

Request

```
PUT /documents/<documentId>/reports/<reportId>/elements/<elementId>
```

Response

Response type: `application/xml` or `application/json`

Example (XML)

Creating and assigning a custom format to a table cell using a template

```
PUT /documents/22036/reports/12/elements/23
```

Request body:

```
<element>
    <content>
        <expression>
            <format type="Currency">
                <template positive="'¥'#,##0[Green]" negative="-'¥'#,##0[Red]"/>
            </format>
        </expression>
    </content>
</element>
```

Response body:

```
<success>
    <message>The resource of type "Report element" with identifier "23" has been
successfully updated.</message>
    <id>23</id>
</success>
```

Creating and assigning a custom format to a table cell using a token

`PUT /documents/22036/reports/12/elements/23`

Request body:

```
<element>
    <content>
        <expression>
            <format>

<token>L[$]T[#]T[gs]T[#]T[#]T[0]T[ds]T[0]T[0]T[Cb]T[FS]L[$]T[-]T[#]T[gs]T[#]T[#
]T[0]T[ds]T[0]T[0]T[Cr]T[FS]T[Ck]T[FS]T[Cgr]</token>
            </format>
        </expression>
    </content>
</element>
```

Response body:

```
<success>
    <message>The resource of type "Report element" with identifier "23" has been
successfully updated.</message>
    <id>23</id>
</success>
```

8.1.13.4 Getting the Font Mappings

Usage

Gets a list of all declared font mappings on a Web Intelligence server. There are three available platforms: HTML, Java, and Windows, with different font mappings. This call returns all mappings for all platforms.

Request

`GET /configuration/fontmappings`

Response

Response type: `application/xml` or `application/json`

Response body: the list of all declared font mappings on the Web Intelligence server:

- The `platform` attribute specifies the type of platform that uses the fonts
- The `<serverName>` element is the name of the font server
- The `<platformName>` element lists the available fonts. Fonts with more than one word as their name are enclosed in quotes.

Example

```
<fontmappings>
    <mapping platform="HTML">
        <font>
            <serverName>Arial</serverName>
            <platformName>Arial, Helvetica, 'Courier New', 'Times New Roman'</
platformName>
        </font>
        <font>
            <serverName>BOJapan</serverName>
            <platformName>'MS Gothic', 'Arial Unicode MS'</platformName>
        </font>
    </mapping>
    <mapping platform="Java">
        <font>
            <serverName>Arial</serverName>
            <platformName>Arial, Helvetica, 'Courier New', 'Times New Roman'</
platformName>
        </font>
    </mapping>
    <mapping platform="Windows">
        <font>
            <serverName>Arial</serverName>
            <platformName>Arial</platformName>
        </font>
        <font>
            <serverName>default</serverName>
            <platformName>Arial</platformName>
        </font>
    </mapping>
</fontmappings>
```

8.1.13.5 Getting the Locales

Usage

Retrieves the list of locales from the server.

Request

```
GET /configuration/locales
```

Response

Response type: `application/xml` or `application/json`

Example (XML)

```
GET /configuration/locales
```

Response:

```
<locales>
    <locale>
        <id>it_CH</id>
        <name>Italian (Switzerland)</name>
    </locale>
    <locale>
        <id>pt_BR</id>
        <name>Portuguese (Brazil)</name>
    </locale>
    ...
    <locale>
        <id>es_PY</id>
        <name>Spanish (Paraguay)</name>
    </locale>
</locales>
```

8.1.13.6 Getting the Report Skins

Usage

Gets a list of all skins of all types declared on a Web Intelligence Server.

Request

```
GET /configuration/skins
```

Response

Response type: `application/xml` or `application/json`

Response body: details of the available skins identified by:

```
<skin type="Cell|Block|Section|Report">
    <name>
    <background>
        <color>
        <image>
```

Where:

- `<name>` is the unique name of the skin
- `<background>` is the background fill for the skin: a color, pattern, or a gif URL.

Example

```
<skins>
    <skin type="Cell">
        <name>Curve</name>
        <background>
            <color rgb="#ffffff"/>
            <image src="boimg://skin_marble.gif" display="Tile"/>
        </background>
    </skin>
    <skin type="Cell">
        <name>Business Objects</name>
        <background>
            <color rgb="#ffffff"/>
            <image src="boimg://skin_bo.gif" display="Stretch"/>
        </background>
    </skin>
    <skin type="Block">
        <name>Business Objects</name>
        <background>
            <color rgb="#ffffff"/>
            <image src="boimg://skin_bo.gif" display="Stretch"/>
        </background>
    </skin>
    <skin type="Section">
        <name>Business Objects</name>
        <background>
            <color rgb="#ffffff"/>
            <image src="boimg://skin_bo.gif" display="Stretch"/>
        </background>
    </skin>
    <skin type="Section">
        <name>Dots</name>
        <background>
            <color rgb="#ffffff"/>
            <image src="boimg://skin_listing.gif" display="Tile"/>
        </background>
    </skin>
    <skin type="Report">
        <name>Dots</name>
        <background>
            <color rgb="#ffffff"/>
            <image src="boimg://skin_listing.gif" display="Tile"/>
```

```
        </background>
    </skin>
</skins>
```

8.1.13.7 Getting the Styles of a Document

Usage

Gets the list of all defined styles in a given document.

Request

`GET /documents/<documentID>/styles?unit=<unit>`

Where:

- `<unit>` is an optional parameter of type string that defines the unit of measurement used for all dimensional values such as background width and height. Values are `metric` (default) , `inch` and `centimeter`.

Response

Response type: `application/xml` or `application/json`

Response body: a series of `<style>` elements.

Example

`GET /documents/3422/styles`

```
<styles>
    <style>
        <id>1</id>
        <background>
            <color rgb="#ffffff"/>
        </background>
        <alignment horizontal="Left" vertical="Bottom"/>
    </style>
    <style>
        <id>2</id>
        <parentId>1</parentId>
        <border>
            <top thickness="None" rgb="#000000" style="None"/>
            <bottom thickness="Thin" rgb="#000000" style="Plain"/>
            <left thickness="None" rgb="#000000" style="None"/>
            <right thickness="None" rgb="#000000" style="None"/>
```

```
        </border>
        <font size="12" face="Arial" italic="false" bold="true"
strikethrough="false"
            underline="false" rgb="#000000"/>
        <alignment horizontal="Left" vertical="Bottom"/>
    </style>
    <style>
        <id>3</id>
        <parentId>1</parentId>
        <border>
            <top thickness="Thin" rgb="#cacad9" style="Plain"/>
            <bottom thickness="Thin" rgb="#cacad9" style="Plain"/>
            <left thickness="Thin" rgb="#cacad9" style="Plain"/>
            <right thickness="Thin" rgb="#cacad9" style="Plain"/>
        </border>
        <background>
            <color rgb="#5175b9"/>
        </background>
        <font size="9" face="Arial" italic="false" bold="true"
strikethrough="false"
            underline="false" rgb="#ffffff"/>
        <alignment horizontal="Left" vertical="Bottom"/>
    <style>
</styles>
```

Related Information

Getting the List of Documents [page 226]

8.1.13.8 Getting the Details of a Style

Usage

Gets the description of a style of a Web Intelligence document.

Request

`GET /documents/<documentID>/styles/<styleID>?unit=<unit>`

Where:

- `<unit>` is an optional parameter of type string that defines the unit of measurement used for all dimensional values such as background width and height. Values are `metric` (default) , `inch` and `centimeter`.

Response

Response type: `application/xml` or `application/json`

Response body: details of a document style identified by `<style>`

Example

`GET /documents/5022/styles/3`

```
<style>
   <id>3</id>
   <parentId>58</parentId>
     <border>
         <top thickness="thin" rgb="#cacad9" style="plain"/>
         <bottom thickness="thin" rgb="#cacad9" style="plain"/>
         <left thickness="thin" rgb="#cacad9" style="plain"/>
         <right thickness="thin" rgb="#cacad9" style="plain"/>
     </border>
     <background>
         <color rgb="#5175b9"/>
     </background>
     <font size="9" face="Arial" italic="false" bold="true" strikethrough="false"
         underline="false" rgb="#ffffff"/>
     <alignment horizontal="left" vertical="bottom"/>
</style>
```

Related Information

Getting the List of Documents [page 226]
Getting the Styles of a Document [page 262]

8.1.13.9 Adding a Style to a Document

Usage

Adds a style to a Web Intelligence document referenced by its ID.

Request

`POST /documents/<documentID>/styles?unit=<unit>`

Where:

- `<unit>` is an optional parameter of type string that defines the unit of measurement used for all dimensional values such as background width and height. Values are `metric` (default) , `inch` and `centimeter`.

Request body:

```
<style>
    <border>
    <background>
    <font>
    <alignment>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`POST /documents/3422/styles?unit=inch`

Request body:

```
<style>
    <background width="2.3" height="1.6"/>
</style>
```

Response:

```
<success>
    <message>The resource of type "Style" with identifier "64" has been
successfully created.</message>
    <id>64</id>
</success>
```

Related Information

Getting the List of Documents [page 226]

8.1.13.10 Updating a Style

Usage

Updates a style of a Web Intelligence document.

Request

`PUT /documents/<documentID>/styles/<styleID>?unit=<unit>`

Where:

- `<unit>` is an optional parameter of type string that defines the unit of measurement used for all dimensional values such as background width and height. Values are `metric` (default) , `inch` and `centimeter`.

Request body:

```
<style>
    <id>
    <parentId>
    <border>
    <background>
    <font>
    <alignment>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`PUT /documents/5022/styles/24`

Request body:

```
<style>
   <id>3</id>
   <parentId>58</parentId>
     <border>
         <top thickness="thin" rgb="#cacad9" style="plain"/>
         <bottom thickness="thin" rgb="#cacad9" style="plain"/>
         <left thickness="thin" rgb="#cacad9" style="plain"/>
         <right thickness="thin" rgb="#cacad9" style="plain"/>
     </border>
     <background>
```

```
        <color rgb="#5175b9"/>
    </background>
    <font size="9" face="Arial" italic="false" bold="true" strikethrough="false"
        underline="false" rgb="#ffffff"/>
    <alignment horizontal="left" vertical="bottom"/>
</style>
```

Response:

```
<success>
    <message>The resource of type "Style" with identifier "24" has been
successfully updated.</message>
    <id>24</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the Styles of a Document [page 262]

8.1.13.11 Deleting a Style

Usage

Deletes a style from a Web Intelligence document.

Request

```
DELETE /documents/<documentID>/styles/<styleID>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`DELETE /documents/5022/styles/L4`

```
<success>
    <message>The resource of type "Style" with identifier "L4" has been
successfully removed.</message>
    <id>L4</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the Styles of a Document [page 262]

8.1.13.12 Getting the CSS of a Document

Usage

Returns the CSS attached to a Web Intelligence document.

Request

`GET /documents/<documentID>/css`

Response

Response type: `text/css`

Example

`GET /documents/5022/css`

Response:

```
FORM CELL.ia-form-header {    /* Settings for header cells in a form  */
      background-fill:color;
      font-size:9pt;
      color:#1D7DB3;
```

```
      background-color:#f8fbfc;
      font-weight-bold:yes;
      never-alternate:yes;
}
CELL.ia-form-separator {    /* Settings for cells separating two form instances
*/
      background-fill:none;    /* Invisible cell making the separation */
      border-top-style:none;
      border-right-style:none;
      border-bottom-style:none;
      border-left-style:none;
      border-top-width:0;
      border-right-width:0;
      border-bottom-width:0;
      border-left-width:0;
}
```

8.1.13.13 Updating the CSS of a Document

Usage

Updates the CSS attached to a Web Intelligence document.

Request

`PUT /documents/<documentID>/css`

Request type: `text/css`

Request body: a CSS file

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
curl -i -X "PUT" -H "accept:application/xml" -H "content-type:text/css" -H X-SAP-
LogonToken:"""%tokenValue%"""
  -d "@css2.xml" "http://localhost:6405/biprws/raylight/v1/documents/5022/css"
```

Request body described in file `css2.xml`:

```
FORM CELL.ia-form-header {    /* Settings for header cells in a form  */
```

```
      background-fill:color;
      font-size:9pt;
      color:#1D7DB3;
      background-color:#f8fbfc;
      font-weight-bold:yes;
      never-alternate:yes;
}
CELL.ia-form-separator {    /* Settings for cells separating two form instances
*/
      background-fill:none;    /* Invisible cell making the separation */
      border-top-style:none;
      border-right-style:none;
      border-bottom-style:none;
      border-left-style:none;
      border-top-width:0;
      border-right-width:0;
      border-bottom-width:0;
      border-left-width:0;
}
```

Response:

```
<success>
    <message>The resource of type "Document" with identifier "5022" has been
successfully updated.</message>
    <id>5022</id>
</success>
```

8.1.13.14 Getting the Charsets

Usage

Gets a list of all supported charsets on a Web Intelligence Server. Charset is used as a parameter when exporting a document as CSV format.

Request

`GET /configuration/charsets`

Response

Response type: `application/xml` or `application/json`

Response body: details of the supported charsets, where:

- `<charset>` contains the description of a supported charset
- `<name>` is the charset name
- `<description>` is the charset description

Example

```
<charsets>
    <charset>
        <name>EUC-KR</name>
        <description>Korean (EUC-KR)</description>
    </charset>
    <charset>
        <name>ISO-8859-8</name>
        <description>Hebrew (ISO-8859-8)</description>
    </charset>
    <charset>
        <name>ISO-8859-5</name>
        <description>Cyrillic (ISO-8859-5)</description>
    </charset>
    . . .
    <charset>
        <name>UTF-8</name>
        <description>UTF-8</description>
    </charset>
    <charset>
        <name>HZ-GB-2312</name>
        <description>Chinese Simplified (HZ-GB2312)</description>
    </charset>
    <charset>
        <name>CNS-11643</name>
        <description>Chinese Traditional (EUC-TW)</description>
    </charset>
    <charset>
        <name>CP437</name>
        <description>IBM Latin US (CP437)</description>
    </charset>
</charsets>
```

8.1.13.15 Getting the Chart Types

Usage

Gets the list of available chart types.

Request

`GET /configuration/visualizations`

Response

Response type: `application/xml` or `application/json`

Response body: details of the supported chart type, where:

- `<visualization type>` contains the description of a supported chart type Tree map, Heat map, Pie chart, Tag cloud and so on.
- `<visualization kind>` specifies whether the chart is built-in or a custom element.
- `<name>` is the chart type name
- `<description>` is the description of the chart type
- <serverID> is the identifier of the delegated rendering server
- `<category>` describes the category of chart. For example, a Tree map and a Heat map are in the Map category.

Example

```
<visualizations>
    <visualization type="TreeMap" kind="Builtin">
        <name>Tree Map</name>
        <description>This chart displays values within nested rectangles that
can be colored.
            The levels of nesting correspond to the level of hierarchical
breakdown.
            The size of the rectangles and their color both express a set of
values.</description>
        <category>Map</category>
    </visualization>
    <visualization type="HeatMap" kind="Builtin">
        <name>Heat Map</name>
        <description>This map displays values that are represented by colors in
a map using
             a category axis and optionally a second category axis.
             The colors of the rectangles are determined by a measure value.</
description>
        <category>Map</category>
    </visualization>
    <visualization type="Pie" kind="Builtin">
        <name>Pie Chart</name>
        <description>A circular chart made up of sectors. The area of the circle
represents a whole,
             and the sectors of the circle represent the parts of a whole.</
description>
        <category>Pie</category>
    </visualization>
    . . .
    <visualization type="TagCloud" kind="Builtin">
        <name>Tag Cloud</name>
        <description>A mono-dimensional visualization representing data as words
where the word
             font size represents its relative weight in the dataset.</
description>
        <category>Tag Cloud</category>
    </visualization>
    <visualization type="mekko" kind="Custom">
        <name>CVOM HTML5 mekko chart</name>
        <description>Description for Mekko</description>
        <serverId>Delegated_Rendering_Server</serverId>
        <category>Delegated_Rendering_Server</category>
    </visualization>
    <visualization type="google" kind="Custom">
        <name>Google Maps.</name>
        <serverId>Delegated_Rendering_Server</serverId>
        <category>Delegated_Rendering_Server</category>
    </visualization>
```

```
</visualizations>
```

Related Information

Chart Types [page 69]

8.1.13.16 Getting the Gradient Orientations

Usage

Retrieves the list of gradient orientations that can be used in a gradient color.

Request

```
GET /configuration/gradientorientations
```

Response

Response type: `application/xml` or `application/json`

Example (XML)

```
GET /configuration/gradientorientations
```

Response:

```
<gradientorientations>
    <gradientorientation>
        <id>HORIZONTAL</id>
        <name>Horizontal</name>
    </gradientorientation>
    <gradientorientation>
        <id>HORIZONTAL_INVERSE</id>
        <name>Horizontal inverted</name>
    </gradientorientation>
    <gradientorientation>
        <id>VERTICAL</id>
        <name>Vertical</name>
```

```
    </gradientorientation>
    <gradientorientation>
        <id>VERTICAL_INVERSE</id>
        <name>Vertical inverted</name>
    </gradientorientation>
    <gradientorientation>
        <id>DIAGONAL_UP</id>
        <name>Diagonal up</name>
    </gradientorientation>
    <gradientorientation>
        <id>DIAGONAL_UP_INVERSE</id>
        <name>Diagonal up inverted</name>
    </gradientorientation>
    <gradientorientation>
        <id>DIAGONAL_DOWN</id>
        <name>Diagonal down</name>
    </gradientorientation>
    <gradientorientation>
        <id>DIAGONAL_DOWN_INVERSE</id>
        <name>Diagonal down inverted</name>
    </gradientorientation>
</gradientorientations>
```

8.1.13.17 Aggregating Configuration Calls

Usage

Returns the results of other configuration calls according to a given template.

Request

`POST /configuration`

Request type: `application/xml` or `application/json`

Response

Response type: `application/xml` or `application/json`

Example (XML)

`POST /configuration`

Request:

```
<configuration>
    <charsets/>
    <settings/>
</configuration>
```

Response:

```
<configuration>
    <charsets>
        <charset>
            <name>MacThai</name>
            <description>Thai (MacThai)</description>
        </charset>
        <charset>
            <name>MacRoman</name>
            <description>Western (MacRoman)</description>
        </charset>
        ...
    </charsets>
    <settings>
        <property key="WEBI_AUTOSAVE_MAXSIZE_DOCUMENTS">30</property>
        <property key="WEBI_AUTO_MERGE">false</property>
        ...
    </settings>
</configuration>
```

Example (JSON)

`POST /configuration`

Request:

```
{
    "configuration": {
        "charsets": {},
        "fontmappings": {
            "mapping": [{
                    "@platform": "HTML"
                }
            ]
        },
        "formats": {},
        "functions": {},
        "gradientorientations": {},
        "locales": {},
        "markerpalettes": {},
        "operators": {},
        "palettes": {},
        "papersizes": {},
        "settings": {},
        "skins": {},
        "visualizations": {
            "visualization": [{
                    "@kind": "Builtin"
                }
            ]
        }
    }
}
```

Response:

```
{
    "configuration": {
        "charsets": {
            "charset": [{
                    "name": "MacThai",
                    "description": "Thai (MacThai)"
                }, {
                    "name": "MacRoman",
                    "description": "Western (MacRoman)"
                },
                ...
            ]
        },
        "fontmappings": { ... },
        "formats": { ... },
        "functions": { ... },
        "gradientorientations ": { ... },
        "locales": { ... },
        "markerpalettes": { ... },
        "operators": { ... }
        "palettes": { ... }
        "papersizes": { ... }
        "settings": { ... }
        "skins": { ... }
        "visualizations": { ... }
        }
    }
}
```

8.1.14 Working with Color Palettes

A color palette defines a series of colors that can be used in a Web Intelligence document.

You can provide built-in color palettes as in the Web Intelligence user interfaces. You can also provide custom palettes.

Getting the Default Color Palettes [page 276]

Getting all the Marker Palettes [page 278]

Getting the List of Custom Palettes [page 279]

Getting the Details of a Custom Palette [page 280]

Creating a Custom Palette [page 281]

Updating the Definition of a Custom Palette [page 283]

Deleting a Custom Palette [page 283]

8.1.14.1 Getting the Default Color Palettes

Usage

Gets the list of default color palettes. This does not return any custom palettes.

A color is expressed using the RGB color model and an opacity. The RGB color value is the # sign followed by an hexadecimal value. The color opacity value is in the range [0, 255] from opaque to invisible.

Request

```
GET /configuration/palettes
```

Response

Response type: `application/xml` or `application/json`

Response Body	Description	Release

`<palette name="string">`
`<color rgb="#..."`
`alpha="number">`
	The name is the default palette name. The RGB and alpha values define a palette color.	Before 4.1 SP5

`<palette>`
`<id>`
`<colors>`
`<color`
`rgb="#..."`
`alpha="number">`
	`<id>` represents the default palette identifier. The RGB and alpha values define a palette color.	Since 4.1 SP5

Example: Before 4.1 SP5

```
<palettes>
    <palette name="SAP Standard 2011">
        <color rgb="#008fd3" alpha="255"/>
        <color rgb="#99d101" alpha="255"/>
        ...
        <color rgb="#61209a" alpha="255"/>
    </palette>
    ...
    <palette name="black&amp;white">
        <color rgb="#5b5b5b" alpha="255"/>
        <color rgb="#d2d2d2" alpha="255"/>
        ...
        <color rgb="#050505" alpha="255"/>
    </palette>
</palettes>
```

Example: Since 4.1 SP5

```
<palettes>
    <palette>
        <id>SAP Standard 2011</id>
        <colors>
            <color rgb="#008fd3" alpha="255"/>
            <color rgb="#99d101" alpha="255"/>
            ...
            <color rgb="#979ca3" alpha="255"/>
        </colors>
    </palette>
    ...
    <palette>
        <id>black&amp;white</id>
        <colors>
            <color rgb="#5b5b5b" alpha="255"/>
            <color rgb="#d2d2d2" alpha="255"/>
            ...
            <color rgb="#050505" alpha="255"/>
        </colors>
    </palette>
</palettes>
```

8.1.14.2 Getting all the Marker Palettes

Usage

Retrieves the list of marker palettes.

Request

```
GET /configuration/markerpalettes
```

Response

Response type: `application/xml` or `application/json`

Example (XML)

```
GET /configuration/markerpalettes
```

```
<markerpalettes>
    <markerpalette>
        <id>default1</id>
        <symbols>
            <symbol>Square</symbol>
            <symbol>Diamond</symbol>
            <symbol>Star</symbol>
            <symbol>Circle</symbol>
        </symbols>
    </markerpalette>
    <markerpalette>
        <id>default2</id>
        <symbols>
            <symbol>Diamond</symbol>
            <symbol>Circle</symbol>
            <symbol>Square</symbol>
            <symbol>Star</symbol>
        </symbols>
    </markerpalette>
    <markerpalette>
        <id>default3</id>
        <symbols>
            <symbol>Star</symbol>
            <symbol>Square</symbol>
            <symbol>Circle</symbol>
            <symbol>Diamond</symbol>
        </symbols>
    </markerpalette>
    <markerpalette>
        <id>default4</id>
        <symbols>
            <symbol>Circle</symbol>
            <symbol>Star</symbol>
            <symbol>Diamond</symbol>
            <symbol>Square</symbol>
        </symbols>
    </markerpalette>
</markerpalettes>
```

8.1.14.3 Getting the List of Custom Palettes

Usage

Returns the palettes defined by the end-user and attached to a Web Intelligence document.

Request

```
GET /documents/<documentID>/palettes
```

Response

Response type: `application/xml` or `application/json`

The response is the list of custom palettes. A custom palette is defined as follows:

```
<palette>
    <id>
    <name>
    <colors>
        <color>
```

Example

`GET /documents/1234/palettes`

```
<palettes>
    <palette>
        <id>5f95e34e-b5c1-49d5-ac60-eb73ee5527c2</id>
        <name>custom palette 1</name>
        <colors>
            <color rgb="#5b5b5b" alpha="100"/>
            <color rgb="#d2d2d2" alpha="100"/>
            <color rgb="#efefef" alpha="100"/>
            <color rgb="#848484" alpha="100"/>
            <color rgb="#aeadae" alpha="100"/>
            <color rgb="#0f0f0f" alpha="100"/>
            <color rgb="#ffffff" alpha="100"/>
            <color rgb="#bfbfbf" alpha="100"/>
            <color rgb="#6f706f" alpha="100"/>
            <color rgb="#a4a4a4" alpha="100"/>
            ...
        </colors>
    </palette>
    <palette>
        <id>9dccd8a9-ac02-4301-828a-7b0cc81bb937</id>
        <name>custom palette 2</name>
        <colors>
            <color rgb="#00ff00" alpha="50"/>
            <color rgb="#00fa00" alpha="50"/>
        </colors>
  </palette>
</palettes>
```

8.1.14.4 Getting the Details of a Custom Palette

Usage

Returns the color information of a custom palette attached to a Web Intelligence document.

Request

```
GET /documents/<documentID>/palettes/<paletteID>
```

Response

Response type: `application/xml` or `application/json`

Example

```
GET /documents/1234/palettes/f5d53881-6467-4572-a616-c20bc8d1a687
```

```
<palette>
    <id>f5d53881-6467-4572-a616-c20bc8d1a687</id>
    <name>custom palette 3</name>
    <colors>
        <color rgb="#5b5b5b" alpha="100"/>
        <color rgb="#d2d2d2" alpha="100"/>
        <color rgb="#efefef" alpha="100"/>
        <color rgb="#848484" alpha="100"/>
        <color rgb="#aeadae" alpha="100"/>
        <color rgb="#0f0f0f" alpha="100"/>
        <color rgb="#ffffff" alpha="100"/>
        <color rgb="#bfbfbf" alpha="100"/>
        <color rgb="#6f706f" alpha="100"/>
        <color rgb="#a4a4a4" alpha="100"/>
        ...
    </colors>
</palette>
```

Related Information

Getting the List of Documents [page 226]

8.1.14.5 Creating a Custom Palette

Usage

Adds a custom palette to the current Web Intelligence document.

Request

`POST /documents/<documentID>/palettes`

Request body:

```
<palette>
    <name>
    <colors>
        <color>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request. An ID is assigned to the new custom palette.

Example

`POST /documents/1234/palettes`

Request body:

```
<palette>
  <name>custom palette 2</name>
  <colors>
    <color rgb="#00ff00" alpha="50"/>
    <color rgb="#00fa00" alpha="50"/>
  </colors>
</palette>
```

Response:

```
<success>
    <message>The resource of type "Palette" with identifier "ff6079a2-
df7f-4df6-8278-9c5dd6246b2e" has been successfully created.</message>
    <id>ff6079a2-df7f-4df6-8278-9c5dd6246b2e</id>
</success>
```

Related Information

Getting the List of Documents [page 226]

8.1.14.6 Updating the Definition of a Custom Palette

Usage

Updates the name and color definitions of a custom palette attached to a Web Intelligence document.

Request

`PUT /documents/<documentID>/palettes/<paletteID>`

Request body:

```
<palette>
   <name>
   <color>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`PUT /documents/1234/palettes/ff6079a2-df7f-4df6-8278-9c5dd6246b2e`

```
<success>
    <message>The resource of type "Palette" with identifier "ff6079a2-
df7f-4df6-8278-9c5dd6246b2e" has been successfully updated.</message>
    <id>ff6079a2-df7f-4df6-8278-9c5dd6246b2e</id>
</success>
```

8.1.14.7 Deleting a Custom Palette

Usage

Deletes a custom palette attached to a Web Intelligence document.

Request

```
DELETE /documents/<documentID>/palettes/<paletteID>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/1234/palettes/ac468f5c-2035-42ca-8e6f-76700657fa91
```

```
<success>
    <message>The resource of type "Palette" with identifier
"ac468f5c-2035-42ca-8e6f-76700657fa91" has been successfully removed.</message>
    <id>ac468f5c-2035-42ca-8e6f-76700657fa91</id>
</success>
```

8.1.15 Managing Functions, Operators, and Variables

A series of APIs is provided to perform tasks on formula engine functions and operators, and variables.

Getting the Formula Engine Functions [page 284]

Getting the Formula Engine Operators [page 285]

Getting the Variables of a Document [page 286]

Getting the Definition of a Variable [page 288]

Adding a Variable [page 289]

Editing a Variable [page 292]

Deleting a Variable [page 294]

8.1.15.1 Getting the Formula Engine Functions

Usage

Gets all functions of the available formula engine. This can be used to create formulas in the Report Specification or define variables in the document dictionary.

Request

```
GET /configuration/functions
```

Response

Response type: `application/xml` or `application/json`

Response body: a list of functions, with the following information:

- `<function category="Logical" returnType="Boolean">` Function type
- `<id>`
- `<name>`
- `<description>`
- `<syntax>`

Example

```
<functions>
    <function category="Logical" returnType="Boolean">
        <id>EVEN</id>
        <name>Even</name>
        <description>Determines whether a number is even</description>
        <syntax>bool Even(number)</syntax>
    </function>
    <function category="Document" returnType="String">
        <id>DOCUMENTAUTHOR</id>
        <name>DocumentAuthor</name>
        <description>Returns the InfoView logon of the document creator</
description>
        <syntax>string DocumentAuthor()</syntax>
    </function>
</functions>
```

8.1.15.2 Getting the Formula Engine Operators

Usage

Gets all operators of the formula engine. This can be used to create formulas in the Report Specification or define variables in the document dictionary.

Request

```
GET /configuration/operators
```

Response

Response type: `application/xml` or `application/json`

Response body: details of the document operators available from the formula engine. Each operator is described as follows:

- `<id>`
- `<name>`
- `<description>`
- `<syntax>`

Example

```
<operators>
    <operator>
        <id>LINEAR</id>
        <name>Linear</name>
        <description>Tells the Interpolation function to use linear regression
to supply
            missing measure values</description>
        <syntax>Linear</syntax>
    </operator>
    <operator>
        <id>INLIST</id>
        <name>InList</name>
        <description>Determines if a value is in a list</description>
        <syntax>bool test_value Inlist(value_list)</syntax>
    </operator>
</operators>
```

8.1.15.3 Getting the Variables of a Document

Usage

Gets the content of a variable dictionary of a Web Intelligence document.

Request

```
GET /documents/<documentID>/variables
```

Response

Response type: `application/xml` or `application/json`

Response body: a list of variables used by the document, where:

- `dataType` and `qualification` attributes of `<variable>` are the type and qualification of the variable (measure, attribute or dimension)
- `grouping="true"` attribute is only for grouping variables
- `<id>` is the identifier of the variable
- `<name>` is the name of the variable, as used in the document

Example

```
GET /documents/4326/variables
```

```
<variables>
    <variable dataType="Numeric" qualification="Measure">
        <id>L6</id>
        <name>Min Revenue</name>
    </variable>
    <variable dataType="Numeric" qualification="Measure">
        <id>L7</id>
        <name>RevenueThreshold</name>
    </variable>
    <variable dataType="Numeric" qualification="Measure">
        <id>L8</id>
        <name>Threshold factor</name>
    </variable>
    <variable dataType="Numeric" qualification="Measure">
        <id>L9</id>
        <name>Threshold Max</name>
    </variable>
    <variable dataType="Numeric" qualification="Measure">
        <id>LA</id>
        <name>Threshold Min</name>
    </variable>
    <variable grouping="true" dataType="String" qualification="Dimension">
        <id>LB</id>
        <name>Month+</name>
    </variable>
</variables>
```

Related Information

Getting the List of Documents [page 226]

8.1.15.4 Getting the Definition of a Variable

Usage

Gets the definition of a variable from the variable dictionary of a Web Intelligence document.

Request

```
GET /documents/<documentID>/variables/<variableID>
```

Response

Response type: `application/xml` or `application/json`

Response body: definition of the variable, described as follows:

- `dataType` and `qualification` attributes of `<variable>` are the type and qualification of the variable (measure, attribute or dimension)
- `<id>`
- `<name>` is the name of the variable, as used in the document
- `<description>` optional, can be used to describe how the variable is used
- `<formulaLanguageId>`
- `<definition>` is the formula used by the variable

Example: Regular Variable

```
GET /documents/1234/variables/L9
```

```
<variable dataType="Numeric" qualification="Measure">
    <id>L9</id>
    <name>Threshold Max</name>
    <description>This is the maximum threshold.</description>
    <formulaLanguageId>[Threshold Max]</formulaLanguageId>
    <definition>=[RevenueThreshold]*(1+[Threshold factor])</definition>
</variable>
```

Example: Grouping Variable

```
GET /documents/6409/variables/L2
```

```
<variable grouping="true" dataType="String" qualification="Dimension">
    <id>L1</id>
    <name>Month+</name>
    <description>This grouping simplifies the table.</description>
    <formulaLanguageId>[Month+]</formulaLanguageId>
    <dimensionId>DP0.DOb9</dimensionId>
    <groups>
        <group>
            <name>Q1</name>
            <values>
                <value>1</value>
                <value>2</value>
                <value>3</value>
            </values>
        </group>
        <group>
            <name>Q3</name>
            <values>
                <value>7</value>
                <value>8</value>
                <value>9</value>
            </values>
        </group>
        <group ungroupedValues="AutomaticallyGrouped">
            <name>Others</name>
            <values>
                <value>4</value>
                <value>5</value>
                <value>6</value>
                <value>10</value>
                <value>11</value>
                <value>12</value>
            </values>
        </group>
    </groups>
</variable>
```

Related Information

Getting the List of Documents [page 226]
Getting the Variables of a Document [page 286]

8.1.15.5 Adding a Variable

Usage

Adds a regular or grouping variable definition to a Web Intelligence document.

Request

`POST /documents/<documentID>/variables`

Request body in the case of a regular variable:

```
<variable qualification="Measure|Attribute|Dimension">
    <name>
    <description>
    <definition>
```

Where:

- `qualification` is the type of variable
- `<description>` is the description of the variable (optional)
- `<definition>` is the formula used by the variable

> → Remember
>
> The regular variables must adhere to the following rules:
>
> - The formula must be valid.
> - You can only create a measure, an attribute or a dimension.
> - When you create an attribute, the associated dimension is mandatory.

Request body in the case of a grouping variable:

```
<variable grouping="true">
    <description>
    <dimensionId>
    <groups>
        <group>
            <name>
            <values>
                <value>
```

The name of the variable is computed if not provided by the user or in case of name conflict.

> → Remember
>
> The grouping variables must adhere to the following rules:
>
> - The `grouping` attribute must be present and set to `true`.
> - The `<dimensionId>` element should point to a valid attribute or dimension.
> - Only one group can have the `ungroupedValues` attribute. Values assigned to the `ungroupedValues` group are ignored.
> - A group must have a name and contain at least one value, except the `ungroupedValues` group.
> - A value can be assigned to only one group and must belong to the parent list of values.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: Regular Variable

```
POST /documents/4326/variables
```

Request body:

```
<variable qualification="Measure">
    <name>new variable</name>
    <description>your description</description>
    <definition>=[RevenueThreshold]*[Threshold factor]</definition>
</variable>
```

Response:

```
<success>
    <message>The resource of type "Variable" with identifier "LB" has been
successfully created.</message>
    <id>LB</id>
</success>
```

Example: Grouping Variable

```
POST /documents/6409/variables
```

Request body:

```
<variable grouping="true">
    <dimensionId>DP0.DOb9</dimensionId>
    <groups>
        <group>
            <name>From January to April</name>
            <values>
                <value>1</value>
                <value>2</value>
                <value>3</value>
                <value>4</value>
            </values>
        </group>
        <group>
            <name>Summer Holidays</name>
            <values>
                <value>7</value>
                <value>8</value>
            </values>
        </group>
    </groups>
</variable>
```

Response:

```
<success>
    <message>The resource of type "Variable" with identifier "L3" has been
successfully created.</message>
    <id>L3</id>
```

```
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Values of a Data Object [page 596]

8.1.15.6 Editing a Variable

Usage

Modifies a regular or grouping variable from the variable dictionary of a Web Intelligence document.

You can change the qualification, name or definition of a regular variable. You can change the groups and group values of a grouping variable.

Request

`PUT /documents/<documentID>/variables/<variableID>`

Request body in the case of a regular variable:

```
<variable qualification="Measure|Attribute|Dimension">
    <name>
    <description>
    <definition>
```

> → Remember
>
> The regular variables must adhere to the following rules:
>
> - The formula must be valid.
> - When you change the definition of the variable, you must refresh the document to commit your change to the report.

Request body in the case of a grouping variable:

```
<variable grouping="true">
    <groups>
        <group>
            <name>
            <values>
                <value>
```

> → Remember
>
> The grouping variables must adhere to the following rules:

- The `grouping` attribute must be present and set to `true`.
- The `<dimensionId>` element is not required.
- Only one group can have the `ungroupedValues` attribute. Values assigned to the `ungroupedValues` group are ignored.
- A group must have a name and contain at least one value, except the `ungroupedValues` group.
- A value can be assigned to only one group and must belong to the parent list of values.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: Regular Variable

```
PUT /documents/8022/variables/L9
```

Request body:

```
<variable qualification="Measure">
    <name>Updated variable</name>
    <description>updated description</description>
    <definition>=[RevenueThreshold]*[Threshold factor]/10</definition>
</variable>
```

Response:

```
<success>
    <message>The resource of type "Variable" with identifier "L9" has been
successfully updated.</message>
    <id>L9</id>
</success>
```

Example: Grouping Variable

```
PUT /documents/6409/variables/L1
```

Request body:

```
<variable grouping="true">
    <groups>
        <group>
            <name>First Semester</name>
            <values>
                <value>1</value>
                <value>2</value>
                <value>3</value>
                <value>4</value>
```

```
                <value>5</value>
                <value>6</value>
            </values>
        </group>
        <group>
            <name>Second Semester</name>
            <values>
                <value>7</value>
                <value>8</value>
                <value>9</value>
                <value>10</value>
                <value>11</value>
                <value>12</value>
            </values>
        </group>
    </groups>
</variable
```

Response:

```
<success>
    <message>The resource of type "Variable" with identifier "L1" has been
successfully updated.</message>
    <id>L1</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Values of a Data Object [page 596]
Getting the Variables of a Document [page 286]

8.1.15.7 Deleting a Variable

Usage

Deletes a variable from a documents variable dictionary.

Ensure that the variable is no longer used by the queries in the document.

Request

```
DELETE /documents/<documentID>/variables/<variableID>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/8022/variables/L9
```

```
<success>
    <message>The resource of type "Variable" with identifier "L9" has been
successfully deleted.</message>
    <id>L9</id>
</success>
```

8.1.16 Managing Links

> **Note**
>
> Links refer to merged dimensions, which is the term used in SAP BusinessObjects Web Intelligence.

To create or modify a link, the following conditions must be respected:

- You can only link dimensions, attributes and hierarchies.
- The expressions to link must have the same data type.
- Each expression that you link to must be from a different data provider.
- The expressions that you link to must not be already used in another link.
- You cannot link to expressions that have been stripped (no query stripping allowed).
- The expressions must be compatible.

About compatibility:

- All custom formulas and variables will be considered as compatible to any other expressions and then ignored.
- Measures are always compatible with everything.
- If dimension A is linked, then A is compatible with all expressions in the flow lists that contain the link.
- Two dimensions or attributes from different flow lists are incompatible. Except when a dimension in linked, as mentioned above.

Getting the Links of a Document [page 296]

Getting the Details of a Link [page 297]

Adding a Link [page 298]

Editing a Link [page 299]

Deleting a Link [page 300]

8.1.16.1 Getting the Links of a Document

Usage

Gets the content of a documents links dictionary.

Request

```
GET /documents/<documentID>/links
```

Response

Response type: `application/xml` or `application/json`

Response body: a series of `<link>` identified by `<id>` and `<name>`.

Example

```
GET /documents/8022/links
```

```
<links>
    <link dataType="String" qualification="Dimension">
        <id>LB</id>
        <name>Category</name>
    </link>
    <link dataType="String" qualification="Dimension">
        <id>L8</id>
        <name>City</name>
    </link>
    ...
    <link dataType="String" qualification="Dimension">
        <id>L6</id>
        <name>Year</name>
    </link>
</links>
```

Related Information

Getting the List of Documents [page 226]

8.1.16.2 Getting the Details of a Link

Usage

Gets the description of a link of a document.

Request

```
GET /documents/<documentID>/links/<linkID>
```

Response

Response type: `application/xml` or `application/json`

Response body: a link described as follows:

- `<id>`
- `<name>`
- `<description>`
- `<linkedExpressions>`

Example

```
GET /documents/7738/links/L6
```

```
<link dataType="String" qualification="Dimension">
    <id>L6</id>
    <name>Year</name>
    <description>Year 2003 - 2006.</description>
    <dataSourceObjectId>DS0.DObc</dataSourceObjectId>
    <formulaLanguageId>[Year]</formulaLanguageId>
    <linkedExpressions>
        <linkedExpression id="DP0.DObc"/>
        <linkedExpression id="DP1.DObc"/>
    </linkedExpressions>
</link>
```

Related Information

Getting the List of Documents [page 226]
Getting the Links of a Document [page 296]

8.1.16.3 Adding a Link

Usage

Adds a link to the documents expressions dictionary.

Request

`POST /documents/<documentID>/links`

Request body:

```
<link>
    <name>
    <linkedExpressions>
        <linkedExpression id="string"/>
```

Response

Response type:`application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`POST /documents/3422/links`

Request body:

```
<link>
    <name>new link</name>
    <linkedExpressions>
        <linkedExpression id="DP0.DObc"/>
        <linkedExpression id="DP1.DObc"/>
    </linkedExpressions>
</link>
```

Response:

```
<success>
    <message>The resource of type "Link" with identifier "L6" has been
successfully created.</message>
    <id>L6</id>
</success>
```

Related Information

Getting the List of Documents [page 226]

8.1.16.4 Editing a Link

Usage

Modifies a link of a Web Intelligence document.

Request

`PUT /documents/<documentID>/links/<linkID>`

Request body:

```
<link>
    <id>
    <name>
    <description>
    <dataSourceObjectId>
    <formulaLanguageId>
    <linkedExpressions>
        <linkedExpression id="string"/>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`PUT /documents/3422/links/L2`

Request body:

```
<link dataType="String" qualification="Dimension">
    <id>L6</id>
    <name>Year</name>
    <description>Year 2003 - 2006.</description>
    <dataSourceObjectId>DS0.DObc</dataSourceObjectId>
    <formulaLanguageId>[Year]</formulaLanguageId>
    <linkedExpressions>
```

```
        <linkedExpression id="DP0.DObc"/>
        <linkedExpression id="DP1.DObc"/>
    </linkedExpressions>
</link>
```

Response:

```
<success>
   <message>The resource of type "Link" with identifier "L2" has been
successfully updated.</message>
   <id>67</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the Links of a Document [page 296]

8.1.16.5 Deleting a Link

Usage

Deletes a document referenced by its ID.

Request

`DELETE /documents/<documentID>/links/<linkID>`

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`DELETE /documents/8022/links/L4`

```
<success>
```

```
    <message>The resource of type "Link" with identifier "L4" has been
successfully removed.</message>
    <id>L4</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the Links of a Document [page 296]

8.1.17 Managing Alerters and Tracker Settings

Alerters

You use the alerters to track the changes in selected data and to configure the display of changed data. Use the formula language to build your custom alerters for formatting data changes. You can include special calculations based on data changes. For example, you can include a calculation to show the difference between the previous value and the current value of a measure.

Tracking Data Changes

When you track data changes, you select a particular data refresh as a reference point. This data is known as the reference data. When you activate the data tracking, you see your data in relation to the reference data.

Formatting to track changes either automatically or manually. You can track the following types of data change:

- Inserted data
- Deleted data
- Changed data
- Increased values
- Decreased values

In automatic data tracking mode, you always compare the current data with the data before the last refresh. This is achieved by automatically setting the current data as the reference data just before each refresh. The reference data is always one refresh behind the current data. Automatic data tracking is appropriate for scheduled documents when you want to compare the current data with the data before the last refresh.

In manual data tracking mode, you select the reference data. You continue to use this data as a reference point until you update the reference point.

Getting the List of Alerters [page 302]

Getting the Details of an Alerter [page 303]

Getting the List of Alerters for a Report Element [page 304]

Adding an Alerter [page 306]
Editing an Alerter [page 308]
Editing all Alerters of a Report Element [page 310]
Deleting an Alerter [page 311]
Deleting all Alerters of a Report Element [page 312]
Getting the Tracker Settings [page 313]
Creating the Tracker Settings [page 314]
Editing the Tracker Settings [page 316]
Deleting the Tracker Settings [page 317]

8.1.17.1 Getting the List of Alerters

Usage

Gets the list of all alerters defined in a document.

Request

```
GET /documents/<documentID>/alerters
```

Response

Response type: `application/xml` or `application/json`

Response body: a list of alerters, with the following information:

- `<id>`
- `<name>`
- `<description>` (optional)

Example

```
GET /documents/1223/alerters
```

```
<alerters>
    <alerter>
        <id>1</id>
        <name>Sales Revenue</name>
```

```
        <description>Test 1</description>
    </alerter>
    <alerter>
        <id>2</id>
        <name>Profit Margin</name>
        <description>Test 2</description>
    </alerter>
</alerters>
```

Related Information

Getting the List of Documents [page 226]

8.1.17.2 Getting the Details of an Alerter

Usage

Gets the full description of an alerter of a Web Intelligence document.

Request

```
GET /documents/<documentID>/alerters/<alerterID>
```

Response

Response type: `application/xml` or `application/json`

Response body: details of the document alerter identified by:

- `<id>`
- `<name>`
- `<description>` (optional)
- `<rule>`, which is the definition of the rule
- `<action>`, which is the definition of the affect on the document formatting

Example

```
GET /documents/8022/alerters/3
```

```
<alerter>
    <id>1</id>
    <name>Sales Revenue</name>
    <description>Test Raylight</description>
    <rule>
        <conditions>
            <condition expressionId="DP0.DO93" operator="Greater">
                <operand>2000000</operand>
            </condition>
        </conditions>
        <action>
            <data>
                <formula type="HyperLink">test</formula>
                <format type="Custom">
                    <template positive="STANDARD"/>
                </format>
            </data>
            <style>
                <background>
                    <color rgb="#ffff00"/>
                </background>
                <font rgb="#ff0000"/>
            </style>
        </action>
    </rule>
</alerter>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Alerters [page 302]

8.1.17.3 Getting the List of Alerters for a Report Element

Usage

Returns the alerters used by a report element of type cell.

Request

```
GET /documents/<documentId>/report/<reportId>/elements/<elementId>/alerters?
allInfo=<boolean>
```

Where:

- `allInfo` is an optional boolean parameter used to select each aleter and get their details. It's set to `true` by default.

Response

Response type: `application/xml` or `application/json`

Example (XML)

Example with no details

```
GET documents/7032/reports/1/elements/7/alerters
```

Response body:

```
<alerters>
    <alerter>
        <id>1</id>
        <name>Sales Revenue</name>
        <description>Description of Sales revenue</description>
    </alerter>
    <alerter>
        ...
</alerters>
```

Example with all aleters details

```
GET /documents/7032/reports/1/elements/7/alerters?allInfo=true
```

Response body:

```
<alerters>
    <alerter>
        <id>1</id>
        <name>Sales Revenue</name>
        <description>Description of Sales revenue</description>
        <rule>
            <id>1</id>
            <conditions>
                <condition expressionId="DP0.DO93" operator="Greater">
                    <operand>2000000</operand>
                </condition>
            </conditions>
            <action>
                <style>
                    <border>
                        <top thickness="None" rgb="#000000" style="None"/>
```

```
                        <bottom thickness="None" rgb="#000000" style="None"/>
                        <left thickness="None" rgb="#000000" style="None"/>
                        <right thickness="None" rgb="#000000" style="None"/>
                    </border>
                    <background width="0.0" height="0.0">
                        <color rgb="#ffff00"/>
                    </background>
                    <font size="0" face="default" italic="false" bold="false"
strikethrough="false" underline="false" rgb="#ff0000"/>
                    <alignment horizontal="Left" vertical="Bottom"
wrapText="false"/>
                </style>
            </action>
        </rule>
    </alerter>
    <alerter>
        ...
</alerters>
```

Example on a report element that isn't a cell

```
GET /documents/7032/reports/1/elements/2/alerters
```

Response body:

```
HTTP Response Code: 400

<error>
    <error_code>WSR 00101</error_code>
    <message>The resource with identifier "2" is not of type "Cell".</message>
</error>
```

8.1.17.4 Adding an Alerter

Usage

Adds an alerter to a Web Intelligence document.

Request

```
POST /documents/<documentID>/alerters
```

Request body:

```
<alerter>
    <name>
    <description>
    <rule>
        <action>
```

> **Note**
>
> You must provide an action when creating a new alerter. An empty alerter will result in an error.

Response

Response type : `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
POST /documents/1223/alerters
```

Request body:

```
<name>ciomplexalerter</name>
    <description>Adda complexe alerter using Raylight</description>
    <rule>
        <conditions>
            <condition expressionId="DP0.DO93" operator="Greater">
                <operand>10</operand>
                <operand>20</operand>
            </condition>
            <condition expressionId="DP0.DO93" operator="Greater">
                <target expressionId="DP0.DO93" />
            </condition>
        </conditions>
        <action>
            <data>
                <formula type="HyperLink">my hyperlink</formula>
                <format type="Custom" sample="1 234,57">
                    <template positive="STANDARD"/>
                </format>
            </data>
            <style>
                <border>
                    <top thickness="Medium" rgb="#800000" style="Plain"/>
                    <bottom thickness="Medium" rgb="#800000" style="Plain"/>
                    <left thickness="Medium" rgb="#800000" style="Plain"/>
                    <right thickness="Medium" rgb="#800000" style="Plain"/>
                </border>
                <background width="0" height="0">
                    <color rgb="#00ff00"/>
                    <image src="bores://00002">
                        <alignment horizontal="Left" vertical="Top"/>
                    </image>
                </background>
                <font size="12" face="Arial" italic="false" bold="false"
strikethrough="true"
                    underline="true" rgb="#ff6600"/>
                <alignment horizontal="Center" vertical="Center"/>
            </style>
        </action>
    </rule>
    <rule>
        <expression>=&quot;hello&quot;</expression>
    </rule>
</alerter>
```

> **Note**
>
> The font size is expressed in "point".

Response:

```
<success>
    <message>The resource of type "Alerter" with identifier "2" has been
successfully created.</message>
    <id>2</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Alerters [page 302]

8.1.17.5 Editing an Alerter

Usage

Updates the definition of an alerter of a Web Intelligence document.

Request

`PUT /documents/<documentID>/alerters/<alerterID>`

Request body:

```
<alerter>
    <id>
    <name>
    <description>
    <rule>
        <action>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/5022/alerters/2
```

Request body:

```
<alerter>
    <name>Sample 2</name>
    <description>Add a complex alerter</description>
    <rule>
        <conditions>
            <condition expressionId="DP0.DO93" operator="Greater">
                <operand>20</operand>
            </condition>
            <condition expressionId="DP0.DO93" operator="Greater">
                <target expressionId="DP0.DO93" />
            </condition>
        </conditions>
        <action>
            <data>
                <formula type="HyperLink">my hyperlink</formula>
                <format type="Custom" sample="1 234,57">
                    <template positive="STANDARD"/>
                </format>
            </data>
            <style>
                <border>
                    <top thickness="Medium" rgb="#800000" style="Plain"/>
                    <bottom thickness="Medium" rgb="#800000" style="Plain"/>
                    <left thickness="Medium" rgb="#800000" style="Plain"/>
                    <right thickness="Medium" rgb="#800000" style="Plain"/>
                </border>
                <background width="0" height="0">
                    <color rgb="#00ff00"/>
                    <image src="bores://00002">
                        <alignment horizontal="Left" vertical="Top"/>
                    </image>
                </background>
                <font size="12" face="Arial" italic="false" bold="false"
strikethrough="true"
                    underline="true" rgb="#ff6600"/>
                <alignment horizontal="Center" vertical="Center"/>
            </style>
        </action>
    </rule>
    <rule>
        <expression>=&quot;hello&quot;</expression>
    </rule>
</alerter>
```

Response:

```
<success>
    <message>The resource of type "Alerter" with identifier "2" has been
successfully updated.</message>
    <id>2</id>
</success>
```

Related Information

Getting the List of Documents [page 226]

Getting the List of Alerters [page 302]

8.1.17.6 Editing all Alerters of a Report Element

Usage

Adds, removes, or reorders the alerters used by a report element of type cell.

Request

```
PUT /documents/<documentId>/report/<reportId>/elements/<elementId>/alerters
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example (XML)

```
PUT /documents/7032/reports/1/elements/7/alerters
```

Request body:

```
<element>
  <content>
    <alerters>
      <id>7</id>
      <id>2</id>
    </alerters>
  </content>
</element>
```

Response body:

```
<success>
    <message>The resource of type "Report element" with identifier "7" has been
successfully updated.</message>
    <id>7</id>
</success>
```

Example with reference to an unknown alerter

```
PUT /documents/7032/reports/1/elements/7/alerters
```

Request body:

```
<element>
  <content>
    <alerters>
      <id>123456</id>
    </alerters>
  </content>
</element>
```

Response body:

```
HTTP Response Code: 404

<error>
    <error_code>WSR 00400</error_code>
    <message>The resource of type "Alerter" with identifier "123456" does not
exist.</message>
</error>
```

8.1.17.7 Deleting an Alerter

Usage

Deletes an alerter.

Request

```
DELETE /documents/<documentID>/alerters/<alerterID>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/5022/alerters/3
```

```
<success>
```

```
    <message>The resource of type "Alerter" with identifier "3" has been
successfully removed.</message>
    <id>3</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Alerters [page 302]

8.1.17.8 Deleting all Alerters of a Report Element

Usage

Removes all alerters used by a report element of type cell.

> **Note**
> The alerters aren't removed from the document.

Request

```
DELETE /documents/<documentId>/report/<reportId>/elements/<elementId>/alerters
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example (XML)

```
DELETE /documents/7032/reports/1/elements/7/alerters
```

Response body:

```
<success>
    <message>The resource of type "Report element" with identifier "7" has been
successfully updated.</message>
    <id>7</id>
</success>
```

8.1.17.9 Getting the Tracker Settings

Usage

Obtains the current trackdata definition for a document. Returns an error if the trackdata function is not activated on this document.

Request

`GET /documents/<documentID>/tracker`

Response

Response type: `application/xml` or `application/json`

Response body: definition of the trackdata information, described as follows:

- The `mode` attribute of `<tracker>` is the mode of Track Data changes:
  - `Auto`: display changes are based on comparison with last data refresh
  - `Manual`: display changes are based on comparison with data refresh from user reference
- The `threshold` attribute specifies the display changes threshold in percent for numeric data. The attribute type is double.

Example

`GET /documents/8022/tracker`

```
<tracker mode="Manual" referenceDate="2012-11-16T10:56:21.951+01:00">
    <added active="true">
        <background/>
        <style italic="false" bold="true" strikethrough="false"
underline="false" rgb="#0000aa"/>
    </added>
    <changed active="true">
        <background rgb="#aabbcc"/>
```

```
        <style italic="false" bold="true" strikethrough="false"
underline="false" rgb="#0000bb"/>
    </changed>
    <removed active="true">
        <background/>
        <style italic="false" bold="true" strikethrough="false"
underline="false" rgb="#0000cc"/>
    </removed>
    <increasing threshold="5.0" applyThreshold="false" active="true">
        <background rgb="#33cc33"/>
        <style italic="false" bold="true" strikethrough="false"
underline="false"/>
    </increasing>
    <decreasing threshold="8.0" applyThreshold="true" active="false">
        <background rgb="#33cc33"/>
        <style italic="true" bold="false" strikethrough="true" underline="true"/>
    </decreasing>
</tracker>
```

Related Information

Getting the List of Documents [page 226]

8.1.17.10 Creating the Tracker Settings

Usage

Activates a trackdata function on a document.

The request body is optional. If the request body is not provided, the trackdata will be activated with default settings or the previous settings if the trackdata function has been activated earlier.

Request

```
POST /documents/<documentID>/tracker
```

Request body:

```
<tracker mode="Manual" referenceDate="DateTime">
    <added>
    <changed>
    <removed>
    <increasing>
    <decreasing>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`POST /documents/8022/tracker`

Request body:

```
<tracker mode="Manual" referenceDate="2012-11-16T10:56:21.951+01:00">
    <added active="true">
        <background/>
        <style italic="false" bold="true" strikethrough="false"
underline="false" rgb="#0000aa"/>
    </added>
    <changed active="true">
        <background rgb="#aabbcc"/>
        <style italic="false" bold="true" strikethrough="false"
underline="false" rgb="#0000bb"/>
    </changed>
    <removed active="true">
        <background/>
        <style italic="false" bold="true" strikethrough="false"
underline="false" rgb="#0000cc"/>
    </removed>
    <increasing threshold="5.0" applyThreshold="false" active="true">
        <background rgb="#33cc33"/>
        <style italic="false" bold="true" strikethrough="false"
underline="false"/>
    </increasing>
    <decreasing threshold="8.0" applyThreshold="true" active="false">
        <background rgb="#33cc33"/>
        <style italic="true" bold="false" strikethrough="true" underline="true"/>
    </decreasing>
</tracker>
```

Response:

```
<success>
    <message> The resource of type "Tracker" has been successfully created for
the document.</message>
    <id>9917</id>
</success>
```

Related Information

Getting the List of Documents [page 226]

8.1.17.11 Editing the Tracker Settings

Usage

Updates a trackdata function on a document.

It updates the style applied on alerters. It is also possible to enable/disable the visualization of a change type. If the track data definition is not provided, the mode is set to `Manual` and the document is set as reference.

The request body is optional. If not provided, the trackdata will be activated with default settings or the previous settings if trackdata has been activated previously.

Request

`PUT /documents/<documentID>/tracker`

Request body:

```
<tracker mode="Manual" referenceDate="DateTime">
    <added>
    <changed>
    <removed>
    <increasing>
    <decreasing>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`PUT /documents/8022/tracker`

Request body in an XML file:

```
<tracker mode="Manual" referenceDate="2012-11-16T10:56:21.951+01:00">
    <added active="true">
        <style italic="false" bold="true" strikethrough="false"
underline="false" rgb="#0000aa"/>
    </added>
    <changed active="true">
        <background rgb="#aabbcc"/>
        <style italic="false" bold="true" strikethrough="false"
underline="false" rgb="#0000bb"/>
    </changed>
    <removed active="true">
```

```
        <style italic="false" bold="true" strikethrough="false"
underline="false" rgb="#0000cc"/>
    </removed>
    <increasing threshold="5.0" applyThreshold="false" active="true">
        <background rgb="#33cc33"/>
        <style italic="false" bold="true" strikethrough="false"
underline="false"/>
    </increasing>
    <decreasing threshold="8.0" applyThreshold="true" active="false">
        <background rgb="#33cc33"/>
        <style italic="true" bold="false" strikethrough="true" underline="true"/>
    </decreasing>
</tracker>
```

Response:

```
<success>
    <message> The resource of type "Tracker" has been successfully updated for
the document.</message>
    <id>9917</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the Tracker Settings [page 313]

8.1.17.12 Deleting the Tracker Settings

Usage

Deletes the trackdata function on the document.

Request

`DELETE /documents/<documentID>/tracker`

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/5022/tracker
```

```
<success>
    <message>The resource of type "Tracker" has been successfully removed.</
message>
    <id>9917</id>
</success>
```

Related Information

Getting the List of Documents [page 226]

8.1.18 Managing Attachments

REST APIs are provided to allow tasks on the attachments of a report.

Adding an Attachment [page 318]

Getting the List of Attachments [page 320]

8.1.18.1 Adding an Attachment

Usage

Adds an attachment to a document.

Request

```
POST /documents/<documentID>/attachments
```

Request type: `multipart/form-data`

Request body:

```
--------------------####boundary123456798
Content-Disposition: form-data; name=""attachmentInfos""
Content-Type: application/xml
<attachment>
    <name>sales_report</name>
    <mimeType>image/png</mimeType> <!--Optional-->
    <size>123</size>
</attachment>
```

```
--------------------####boundary123456798
Content-Disposition: form-data; name=""attachmentContent"";
filename=""logo_picture.png""
Content-Type: image/png
(Content of file)
--------------------####boundary123456798
```

The multipart body request allows at least two parts:

- The information about the attachment (`attachmentInfos`)
- The attachment content (`attachmentContent`)

In case of image upload, the mime types accepted are the following:

- `image/png`
- `image/jpeg`
- `image/gif`
- `image/bmp`

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

CURL command:

```
curl -i -X "POST" -H "accept:application/xml"
-H "content-type:multipart/form-data" -H X-SAP-LogonToken:"""%tokenValue%"" -F
"attachmentInfos=@attachment.xml”
 -F ”attachmentContent=@<file_to_be_uploaded>” http://<serverName>:6405/biprws/
raylight/v1/documents/3422/attachments
```

Request body:

```
<attachment>
    <mimeType>image/png</mimeType>
    <name>myResourceName</name>
    <description>myResourceDescription</description>
</attachment>
```

Response:

```
<success>
    <message>The resource of type "Attachment" has been successfully created.</
message>
    <id>9994</id>
</success>
```

Related Information

Getting the List of Documents [page 226]

8.1.18.2 Getting the List of Attachments

Usage

Gets the list of attachments to a document.

Request

```
GET /documents/<documentID>/attachments
```

Response

Response type: `application/xml` or `application/json`

Response body: details of the attachments of the document. Each attachment is identified by:

- The attachment mime type `<mimeType>`
- Document name `<name>`
- Document description (if one exists, otherwise the tag is empty) `<description>`

Example

```
GET /documents/8022/attachments
```

```
<attachments>
    <attachment>
        <name>barometer.png</name>
        <size>13229</size>
        <releasemode>auto</releasemode>
        <md5hashcode>97B3E3B2745595A2CBA42CA825CCD656</md5hashcode>
        <mimeType>image/png</mime>
    </attachment>
    <attachment>
        <name>statistics.jpg</name>
        <size>18239</size>
        <releasemode>auto</releasemode>
        <md5hashcode>97B3E3B2745595A2CBA42CA825CCD656</md5hashcode>
        <mimeType>image/jpeg</mimeType>
    </attachment>
```

```
</attachments>
```

Related Information

Getting the List of Documents [page 226]

8.1.19 Working with Snapshots

A snapshot is a report of a document that contains the results of a drill action. A snapshot is identified by a snapshot ID token. A document can have the following states:

- Open
- Set Prompts
- Refresh
- Output

These states are affected by each change of the document (not only in the case of a prompt workflow like above). This means for example:

- Drill workflows
- Turn to
- Report filter manipulation
- Fold/unfold
  Storage token can be used in scheduling workflows.

  Creating a Document Snapshot [page 321]

  Getting the List of Snapshots [page 322]

  Restoring a Document to a Snapshot [page 323]

  Deleting a Document Snapshot [page 324]

  Deleting All Snapshots [page 325]

8.1.19.1 Creating a Document Snapshot

Usage

Creates a snapshot of a Web Intelligence document. The call returns the identifier of the newly created storage token.

You can customize one step and go back and forth between initial state and resulting state. This can be used as undo/redo or implement custom prompt dialog boxes.

Request

```
POST /documents/<documentID>/snapshots
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
POST /documents/8022/snapshots
```

```
<success>
    <message>The resource of type "Snapshot" with identifier
"we00000000e3667716ec10" has been successfully created.</message>
    <id>we00000000e3667716ec10</id>
</success>
```

Related Information

Getting the List of Documents [page 226]

8.1.19.2 Getting the List of Snapshots

Usage

Gets the list of snapshot identifiers for a specified document.

Request

```
GET /documents/<documentID>/snapshots
```

Response

Response type: `application/xml` or `application/json`

Response body: the list of snapshot IDs for the document. The `id` attribute of `<snaphot>` is the snapshot identifier.

Example

`GET /documents/8022/snapshots`

```
<snapshots maxStackSize="10">
    <snapshot id="we00000000d447acb133a0"/>
    <snapshot id="we00000000e5df6062ca2a"/>
    <snapshot id="we00010000dc73921d7b5d"/>
</snapshots>
```

Related Information

Getting the List of Documents [page 226]

8.1.19.3 Restoring a Document to a Snapshot

Usage

Restores the document to the state corresponding to the given snapshot.

Request

`PUT /documents/<documentID>?snapshotId=<snapshotID>`

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

PUT /documents/5022?snapshotId=we00000000e5df6062ca2a

```
<success>
    <message>The resource of type "Document" with identifier "5022" has been
successfully updated.</message>
    <id>5022</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Snapshots [page 322]

8.1.19.4 Deleting a Document Snapshot

Usage

Deletes a specific snapshot of a Web Intelligence document.

Request

DELETE /documents/<documentID>/snapshots/<snapshotID >

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

DELETE /documents/8022/snapshots/we00000000e3667716ec10

```
<success>
    <message>The resource of type "Snapshot" with identifier
"we00000000e3667716ec10" has been successfully removed.</message>
    <id>we00000000e3667716ec10</id>
```

```
</success>
```

8.1.19.5 Deleting All Snapshots

Usage

Deletes all snapshots of a Web Intelligence document.

Request

`DELETE /documents/<documentID>/snapshots`

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`DELETE /documents/8022/snapshots`

```
<success>
    <message>The resource of type "Snapshot" has been successfully removed.</
message>
</success>
```

If the document has no snapshot, an HTTP 200 response code is returned with the following message:

```
<success>
    <message>Unable to find resources of type "Snapshot".</message>
</success>
```

8.1.20 Working with Occurrences

You can open the same Web Intelligence document several times in memory. Each open document is named occurrence and has its own lifecycle. This means that you can modify, refresh, and analyze an occurrence of the same document independently of the others.

> → Remember
>
> Occurrences are based on the same document. Therefore, the last saved occurrence overwrites all occurrences previously saved in the CMS repository.

About the State of an Occurrence

As a Web Intelligence document, an occurrence can have one of the following states:

State	Description
`Original`	The occurrence has been loaded to the server, but has not been modified. Its state can be changed to `Unused` to release the available memory, closing the document.
`Modified`	The occurrence has been loaded and modified. Its state can be changed to `Unused`. The document is closed to release the available memory.
`Unused`	An unused occurrence is no longer loaded to the WACS server.
`KeepAlive`	Keeps the occurrence open when the user interacts with the document without making any call to the server to prevent a server timeout. **Note** `KeepAlive` is a pseudo-state, and does not change the actual state of the occurrence.

As soon as you open an occurrence, its status becomes `Original`. You can move an occurrence from a `Modified` or `Original` to `Unused` state to discard all document changes and close the document. This releases the memory of the WACS server. The occurrence moves from `Modified` to `Original` when it is saved in the CMS repository.

Using the Web Intelligence RESTful Web Service SDK

Any API that applies to a document can also apply to an occurrence of this document. For example, you can get the details of a report of an occurrence of a document by calling `GET /documents/<documentID>/occurrences/<occurenceID>/reports/<reportID>`.

The occurrence with identifier `0` refers to the document itself. For example, the call `/documents/<documentID>/reports/<reportID>` is the same as `/documents/<documentID>/occurrences/0/reports/<reportID>`.

Creating an Occurrence [page 327]

Getting the List of Occurrences [page 327]

Getting the Details of an Occurrence [page 328]

Updating an Occurrence [page 329]

8.1.20.1 Creating an Occurrence

Usage

Creates an occurrence of a specific Web Intelligence document.

Request

```
POST /documents/<documentID>/occurrences
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
<success>
    <message>The resource of type "Occurrence" with identifier "1" has been
successfully created.</message>
    <id>1</id>
</success>
```

8.1.20.2 Getting the List of Occurrences

Usage

Returns the list of occurrences available for a specific Web Intelligence document.

Request

```
GET /documents/<documentID>/occurrences
```

Response

Response type: `application/xml` or `application/json`

Response body: a series of `<occurrence>` identified by `<id>` and `<state>`.

Example: XML

```
<occurrences>
    <occurrence>
        <id>0</id>
        <state>Original</state>
    </occurrence>
</occurrences>
```

Example: JSON

```
{
 "occurrences":{
     "occurrence":[{
         "id":0,
         "state":"Original"
         }]
     }
}
```

8.1.20.3 Getting the Details of an Occurrence

Usage

Returns the details of a specific occurrence of a Web Intelligence document.

Request

```
GET /documents/<documentID>/occurrences/<occurrenceID>
```

Response

Response type: `application/xml` or `application/json`

Response body: an `<occurrence>` identified by `<id>` and `<state>`.

Example: XML

```
<occurrence>
    <id>0</id>
    <state>Original</state>
</occurrence>
```

Example: JSON

```
{
 "occurrence":{
     "id":0,
     "state":"Original"
     }
}
```

8.1.20.4 Updating an Occurrence

Usage

Changes the state of an occurrence of a Web Intelligence document referenced by its ID.

Request

`PUT /documents/<documentID>/occurrences/<occurrenceID>`

Request body:

```
<occurrence>
    <state>
```

> → Remember
>
> The request body is optional. If `<state>` is present, no other tag is accepted.

Document State Change	Result
From `Original` to `Unused`	The occurrence is not modified and closed.

Document State Change	Result
From `Original` with no body or an empty body	The occurrence is not modified.
From `Modified` to `Unused`	The occurrence is updated and closed.
From `Modified` with no body or an empty body	The occurrence is updated and saved to the CMS repository.

The `KeepAlive` value keeps the occurrence of the document open when the user interacts with the occurence without making any call to the server to prevent a server timeout.

> **Note**
>
> The `KeepAlive` value does not change the state of the occurrence of the document.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: To Close an Unmodified Occurrence

`PUT /documents/8008/occurrences/1`

Request body:

```
<occurrence>
    <state>Unused</state>
</occurrence>
```

Response body:

```
<success>
    <message>The resource of type "Occurrence" with identifier "10082" has not
been modified.</message>
    <id>10082</id>
</success>
```

Example: To Keep an Occurrence Alive

`PUT /documents/1357/occurrences/3`

Request body:

```
<occurrence>
    <state>KeepAlive</state>
</occurrence>
```

Response body

The response depends on the state of the occurrence. If the occurrence is in the `Original` or `Unused` state, the message is the following:

```
<success>
    <message>The resource of type "Occurrence" with identifier "10084" has not
been modified.</message>
    <id>10084</id>
</success>
```

If the occurrence is in the `Modified` state, then the message is the following:

```
<success>
    <message>The resource of type "Occurrence" with identifier "10084" has been
successfully updated.</message>
    <id>10084</id>
</success>
```

If the request fails because the timeout has expired for instance, the call returns the following message:

```
<error>
    <error_code>WSR 00402</error_code>
    <message>An error occurred when keeping the resource of type "Occurrence"
with identifier "10084" alive.</message>
</error>
```

8.1.21 Working with Referenced Cells

A reference is a variable whose definition and content are based on another cell. It is useful whenever you want to leverage data of a cell that has been obtained using a complex formula.

Below are the tasks you can perform on referenced cells:

- Get the list of referenced cells in a document or get the definition of a specific referenced cell.
- Create, update or delete a referenced cell.

8.1.21.1 Getting the List of a Document Referenced Cells

Usage

Lists all available predefined cells of a document.

Request

```
GET /documents/<documentId>/refcells?allInfo=<boolean>
```

Where:

- `allInfo` is an optional boolean parameter that gives details about a referenced cell.

Response

Response type: `application/xml` or `application/json`

Example

Example with no details

```
GET /documents/19323/refcells
```

Response body (XML):

```
<refcells>
    <refcell dataType="Numeric" qualification="Constant">
        <id>L1</id>
        <name>Sales revenues for 2014.</name>
    </refcell>
    <refcell dataType="Numeric" qualification="Constant">
        <id>L2</id>
        <name>Sales revenues for 2015.</name>
    </refcell>
</refcells>
```

Response body (JSON):

```
{
    "refcells": {
        "refcell": [{
                "@dataType": "Numeric",
                "@qualification": "Constant",
                "id": "L1",
                "name": "Sales revenues for 2014."
            }, {
                "@dataType": "Numeric",
                "@qualification": "Constant",
                "id": "L2",
                "name": "Sales revenues for 2015."
            }
        ]
    }
}
```

Example with details

```
GET /documents/19323/refcells?allInfo=true
```

Response body (XML):

```
<refcells>
    <refcell dataType="Numeric" qualification="Constant">
        <id>L1</id>
        <name>Sales revenues for 2014.</name>
        <formulaLanguageId>[Sales revenues for 2014.]</formulaLanguageId>
        <reference>G.8.2</reference>
        <reportId>16</reportId>
    </refcell>
    <refcell dataType="Numeric" qualification="Constant">
        <id>L2</id>
        <name>Sales revenues for 2015.</name>
        <description>This is a description for 'Sales revenues for 2015'.</
description>
        <formulaLanguageId>[Sales revenues for 2015.]</formulaLanguageId>
        <reference>G.8.1</reference>
        <reportId>16</reportId>
    </refcell>
</refcells>
```

Response body (JSON):

```
{
    "refcells": {
        "refcell": [{
                "@dataType": "Numeric",
                "@qualification": "Constant",
                "id": "L1",
                "name": "Sales revenues for 2014.",
                "formulaLanguageId": "[Sales revenues for 2014.]",
                "reference": "G.8.2",
                "reportId": 16
            }, {
                "@dataType": "Numeric",
                "@qualification": "Constant",
                "id": "L2",
                "name": "Sales revenues for 2015.",
                "description": "This is a description for 'Sales revenues for
2015'.",
                "formulaLanguageId": "[Sales revenues for 2015.]",
                "reference": "G.8.1",
                "reportId": 16
            }
        ]
    }
}
```

8.1.21.2 Getting the Definition of a Referenced Cell

Usage

Gets the definition of a referenced cell.

Request

```
GET /documents/<documentId>/refcells/<referencedCellId>
```

Response

Response type: `application/xml` or `application/json`

Example

```
GET raylight/v1/documents/19323/refcells/L2
```

Response body (XML):

```
<refcell dataType="Numeric" qualification="Constant">
    <id>L2</id>
    <name>Sales revenues for 2015.</name>
    <description>This is a description for 'Sales revenues for 2015'.</
description>
    <formulaLanguageId>[Sales revenues for 2015.]</formulaLanguageId>
    <reference>G.8.1</reference>
    <reportId>16</reportId>
</refcell>
```

Response body (JSON):

```
{
    "refcell": {
        "@dataType": "Numeric",
        "@qualification": "Constant",
        "id": "L2",
        "name": "Sales revenues for 2015.",
        "description": "This is a description for 'Sales revenues for 2015'.",
        "formulaLanguageId": "[Sales revenues for 2015.]",
        "reference": "G.8.1",
        "reportId": 16
    }
}
```

8.1.21.3 Adding a New Referenced Cell

Usage

Creates a referenced cell and adds it to the document dictionary, or duplicates an existing referenced cell.

Request

```
POST /documents/<documentId>/refcells?sourceIds=<string>
```

Where:

- `sourceIds` is an optional comma-separated strings list parameter that lists the existing referenced cells identifiers to duplicate. If not present, a valid definition of the new referenced cell must be provided in the request body.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

Creating a referenced cell with a definition in the request body

```
POST /documents/19323/refcells
```

Request body:

```
<refcell>
    <name>refcell example</name>
    <description>This is a description for this referenced cell.</description>
    <reference>G.8.2</reference>
</refcell>
```

Response body:

```
<success>
    <message>The resource of type "Referenced Cell" with identifier "L3" has
been successfully created.</message>
    <id>L3</id>
</success>
```

Copying a referenced cell

```
POST /documents/19323/refcells?sourceIds=L2
```

Response body:

```
<success>
    <message>The resource of type "Referenced Cell" with identifier "L3" has
been successfully created.</message>
    <id>L3</id>
    </success>
```

Copying multiple referenced cells

```
POST /documents/19323/refcells?sourceIds=L1,L2,L1
```

Response body:

```
<success>
    <message>The resource of type "Document" with identifier "19323" has been
successfully updated.</message>
    <id>19323</id>
</success>
```

8.1.21.4 Editing the Definition of a Referenced Cell

Usage

Edits the name or definition of an existing referenced cell.

Request

```
PUT /documents/<documentId>/refcells/<referencedCellId>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example (XML)

```
PUT raylight/v1/documents/19323/refcells/L2
```

Request body:

```
<refcell>
    <description>This is an updated description.</description>
</refcell>
```

Response body:

```
<success>
    <message>The resource of type "Referenced Cell" with identifier "L2" has
been successfully updated.</message>
    <id>L2</id>
</success>
```

8.1.21.5 Deleting a Referenced Cell

Usage

Deletes a referenced cell.

Request

```
DELETE /documents/<documentId>/refcells/<referencedCellId>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example (XML)

```
DELETE raylight/v1/documents/19323/refcells/L1
```

Response body:

```
<success>
    <message>The resource of type "Referenced Cell" with identifier "L1" has
been successfully removed.</message>
    <id>L1</id>
</success>
```

8.1.22 Adding a Cache Entry to a Document

Usage

Adds a cache entry to a document.

Request

```
POST /documents/<documentID>/cache
```

Request body:

```
<cache>
    <output mimeType="application/pdf">
        <locale value="fr_FR"/>
        <locale value="de_DE"/>
    </output>
    <output mimeType="application/pdf">
        <locale value="en" />
    </output>
</cache>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
POST /documents/3422/cache
```

Request body:

```
<cache>
    <output mimeType="application/pdf">
        <locale value="fr_FR"/>
        <locale value="de_DE"/>
    </output>
    <output mimeType="application/pdf">
        <locale value="en"/>
    </output>
</cache>
```

Response:

```
<success>
    <message>The resource of type "Cache" has been successfully updated.</
message>
    <id>3422</id>
</success>
```

8.2 Managing Reports

> **⚠ Restriction**
>
> The following workflows are not supported:
>
> - Retrieving and setting fold/unfold state of a document report
> - Retrieving and setting collapse/expand state of a hierarchy displayed in a document report
> - Text search in a document or report

Creating a Report [page 339]

Getting the List of Reports [page 341]

Getting the Details of a Report [page 342]

Moving a Report [page 344]

Copying a Report [page 345]

Updating the Properties of a Report [page 346]

Deleting a Report [page 347]

Getting the Map of a Report [page 348]

Exporting Reports [page 349]

Shared Element APIs [page 359]
The REST APIs to manage shared elements in the CMS repository, documents, or reports.

Updating the Structure of a Report [page 360]

Managing Data Filters in Reports [page 365]

Getting the Structure of a Report [page 371]

Managing Report Comments [page 372]

8.2.1 Creating a Report

Usage

Creates a report in the given Web Intelligence document.

Request

```
POST /documents/<documentID>/reports
```

Request body (optional):

```
<report>
```

```
    <name>
```

Where:

- `<name>` is a string that specifies the name of the report to be created. If not specified, the service assigns a name to the report automatically.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: XML

`POST /documents/12782/reports`

Response:

```
<success>
    <message>The resource of type "Report" with identifier "2" has been
successfully created.</message>
    <id>2</id>
</success>
```

The name "Report 2" and the ID "2" have been assigned automatically.

Example: JSON

`POST /documents/5152/reports`

Request:

```
{"report":
    {"name":"Chart Report"}
}
```

Response:

```
{"success":
    {"message":"The resource of type \"Report\" with identifier \"6\" has been
successfully created.",
     "id":6}
}
```

The ID "6" has been assigned automatically.

Related Information

Getting the List of Documents [page 226]

8.2.2 Getting the List of Reports

Usage

Gets the list of reports for a Web Intelligence document.

The following items identify a report:

- Id
- Reference

Request

```
GET /documents/<documentID>/reports
```

Response

Response type: `application/xml` or `application/json`

Response body:

- The `<id>`, `<name>`, and `<reference>` of the reports
- The `<showDataChanges>` Boolean value that expresses whether data changes are displayed

Example: XML

```
GET /documents/12782/reports
```

Response:

```
<reports>
    <report>
        <id>1</id>
        <name>Cross Tab View</name>
        <reference>1.RS</reference>
        <showDataChanges>false</showDataChanges>
    </report>
    <report>
        <id>2</id>
```

```
        <name>Simple charting view</name>
        <reference>2.RS</reference>
        <showDataChanges>false</showDataChanges>
    </report>
</reports>
```

Example: JSON

```
GET /documents/5152/reports
```

Response:

```
{"reports":
    {"report":[
        {"id":1,"name":"Basic
Charts","reference":"1.RS","showDataChanges":false},
        {"id":2,"name":"Combination
Chart","reference":"2.RS","showDataChanges":false},
        {"id":3,"name":"Custom Color
Palette","reference":"3.RS","showDataChanges":false},
        {"id":4,"name":"Chart Property
Support","reference":"4.RS","showDataChanges":false},
        {"id":5,"name":"Element
Linking","reference":"5.RS","showDataChanges":false}]
    }
}
```

8.2.3 Getting the Details of a Report

Usage

Gets the details of the report specified by the given identifier.

Since 4.2 SP04, the pagination mode ("QuickDisplay" or "Page") mode is also returned.

Request

```
GET /documents/<documentID>/reports/<reportID>
```

Response

Response type: `application/xml` or `application/json`

Response body:

- The `<id>`, `<name>`, and `<reference>` of the reports
- The `<showDataChanges>` Boolean value that expresses whether data changes are displayed
- The report style and page settings if defined

Example: XML

```
GET /documents/7858/reports/1
```

Response:

```
<report>
    <id>1</id>
    <name>Revenue by City</name>
    <reference>1.RS</reference>
    <showDataChanges>false</showDataChanges>
    <style>
        <hyperLinkColors visited="#000000" link="#0000ff" hover="#000000"
active="#000000"/>
    </style>
    <pageSettings>
        <margins left="2835" right="2835" top="2835" bottom="2835"/>
        <format orientation="Portrait" height="42094" width="29764"/>
        <records vertical="100" horizontal="20"/>
        <scaling factor="100"/>
    </pageSettings>
    <paginationMode>QuickDisplay</paginationMode>
</report>
```

Example: JSON

```
GET /documents/5152/reports/5
```

Response:

```
{"report":
    {"id":5,
     "name":"Element Linking",
     "reference":"5.RS",
     "showDataChanges":false,
     "style":
         {"hyperLinkColors":

{"@visited":"#000000","@link":"#0000ff","@hover":"#000000","@active":"#000000"}
         },
     "pageSettings":
         {"margins":
{"@left":"2835","@right":"2835","@top":"2835","@bottom":"2835"},
          "format":
{"@orientation":"Portrait","@height":"42094","@width":"29764"},
          "records":{"@vertical":"100","@horizontal":"20"},
          "scaling":{"@factor":"100"}
         }
    }
}
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.2.4 Moving a Report

Usage

Moves a report within a Web Intelligence document.

Request

```
PUT /documents/<documentID>/reports?fromId=<fromID>&toId=<toID>
```

Where:

- `<fromID>` is a valid report identifier (before the move)
- `<toID>` is a valid report identifier (after the move)

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/12782/reports?fromId=1&toId=3
```

Response:

```
<success>
    <message>The resource of type "Report" with identifier "1" has been
successfully moved.</message>
    <id>1</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.2.5 Copying a Report

Usage

Copies a report in a given Web Intelligence document.

> ⓘ Note
>
> Since 4.2 SP04 you can specify 'showDataChanges' and 'showFolding' at creation time.

Request

```
POST /documents/<documentID>/reports?fromId=<fromID>&keepDriller=<keepDriller>
```

Where:

- `<fromID>` is a valid identifier of a report
- `<keepDriller>` preserves the driller mode in the copied report if `true` (default). It is only applicable if the source report is in driller mode.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
POST /documents/14308/reports?fromId=2&keepDriller=false
```

Response:

```
<success>
    <message>Resource of type "Report" with identifier "2" has been successfully
created.</message>
    <id>2</id>
```

```
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.2.6 Updating the Properties of a Report

Usage

Updates the properties of a report in the given Web Intelligence document.

Since 4.2 SP04, you can enable/disable the folding in the report. (`showFolding` `showFolding("true","false")`: enable/disable the folding on the report).

Request

`PUT /documents/<documentID>/reports/<reportID>`

Request body:

```
<report>
    <name>
    <showDataChanges>
```

You can modify the report name and the `<showDataChanges>` value (show("true","false") trackdata on the report).

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/9512/reports/67
```

Request body (XML file):

```
<report>
    <name>New report name</name>
    <showDataChanges>true</showDataChanges>
</report>
```

Response:

```
<success>
    <message>The resource of type "Report" with identifier "67" has been
successfully updated.</message>
    <id="67"></id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.2.7 Deleting a Report

Usage

Deletes a report from the given Web Intelligence document.

> → Remember
>
> You cannot undo this operation.

Request

```
DELETE /documents/<documentID>/reports/<reportID>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/7858/reports/9372
```

```
<success>
    <message>The resource of type "report" with identifier "9372" has been
successfully removed.</message>
    <id>9372</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.2.8 Getting the Map of a Report

Usage

Gets the map of a report in a Web Intelligence document.

The map of a report is the list of all the instances of a report element in the document.

Request

`GET /documents/<documentID>/reports/<reportID>/map?reference=<nodeReference>`

Where:

- `<nodeReference>` is a valid reference of a map node (optional)

Response

Response type: `application/xml` or `application/json`

Response body: the node references of the report map.

Since 4.2 SP04, a "final" attribute has been added to each <node> element. If "true" the node is a leaf, else a node with children.

Example: Getting the Full Map References of a Report

```
GET /documents/18809/reports/1/map
```

Response:

```
<map>
    <node reference="1.G.0" name="2004"/>
    <node reference="1.G.1" name="2005"/>
    <node reference="1.G.2" name="2006"/>
</map>
```

Example: Getting the Map Reference of a Node of a Report for 4.2 SP04

```
GET /documents/7525/reports/1/map?reference=1.G.2
```

Response:

```
<map>
    <node name="California" reference="1.5.2.0" final="false"/>
    <node name="Colorado" reference="1.5.2.1" final="false"/>
    <node name="DC" reference="1.5.2.2" final="false"/>
    <node name="Florida" reference="1.5.2.3" final="false"/>
    <node name="Illinois" reference="1.5.2.4" final="false"/>
    <node name="Massachusetts" reference="1.5.2.5" final="false"/>
    <node name="New York" reference="1.5.2.6" final="false"/>
    <node name="Texas" reference="1.5.2.7" final="false"/>
</map>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.2.9 Exporting Reports

You can export:

- An entire report as a single document
- A report in paginated mode (one file per report page)
- One page of a report

> **Note**
>
> Since 4.2 SP4, if drill mode and drill output mode are both activated for at least one of the reports, then drill information is included in `text/xml` output.

> **Note**
>
> Before 4.2 SP4, default value for dpi was set to 300 for PDF and 96 for other output types. Since 4.2 SP4, the default behavior is to keep the server's settings.

> **Note**
>
> Before 4.2 SP4, the default value for Excel data optimization was set to false. Since 4.2 SP4, the default behaviour is to keep the server's settings.

Exporting a Report in Listing Mode [page 350]

Exporting a Report as a Series of Pages [page 353]

Exporting a Page or Range of Pages of a Report [page 355]

8.2.9.1 Exporting a Report in Listing Mode

Usage

Exports the report in one of the following formats:

- HTML
- Zipped HTML
- MHTML (multipart HTML)
- XML
- PDF
- Microsoft Excel 2003
- Microsoft Excel 2007
- CSV
- Plain text

> **Note**
>
> - If you choose the HTML format, the web service generates the image links. Therefore, the logon token must still be valid when the HTML output is displayed.
> - Exports to HTML will be optimized for your browser if you use the User-Agent HTTP header in the REST API call.

Request

```
GET /documents/<documentID>/reports/<reportID>?<optional_parameters>
```

Optional Parameters

Parameter	Description	Supported Formats
`dpi`	Resolution in dots per inch (dpi) for generated charts. Value between 75 and 9600. Default is 300 for PDF format, 96 for all other formats.	All, except XML and CSV
`chartOutputFormat`	Output format for generated chart. Values are: `jpeg`, `bmp`, `gif`, and `png` (default).	HTML, ZIP, and MHTML
`imageUrl`	String used to customize image links when they cannot be reached from the information system.	HTML
`chartOutputFormat`	Generated charts rendering format. type=string, values=" bmp " or " gif " or " png " or " jpeg ", default=" png "	HTML
`fileName`	A string that defines the name of the ZIP file. If set, the HTML output is zipped to a `fileName.zip` file. Maximum string length is 96.	ZIP
`unit`	A string that defines the unit that sizes will be reported in. Values are `"metric"`, `"millimeter"`, `"point"`, and `"pixel"` (default).	XML
`rawValues`	Boolean. Default is `false`. If `true`, the raw values and their types are exported with the formatted values.	XML
`optimized`	Boolean. Default is `false`. If `true`, the generated output is optimized for calculations inside Microsoft Excel.	Microsoft Excel 2003 and Microsoft Excel 2007
`textQualifier`	Character used to surround each column value. Values are `'` or `"`.	CSV
`columnDelimiter`	String that defines a character put between columns. Values are comma (`,`), semi-colon (`;`) or the special string `Tab`.	CSV
`charset`	String that defines a valid server charset retrieved from the list of charsets.	CSV

Customization of Image Links

You may need to customize the source of an image in a report exported as HTML if the image is unreachable by your system:

```
<img border="0"
style="position:absolute;top:0px;left:0px;width:400px;height:300px;"
    alt="3D Column Chart" src="http://dewdfwadept1054.dhcp.wdf.sap.corp:6405/
biprws/
    raylight/v1/documents/6406/reports/1/images/dxXMLDraft.drilloff_1*2*6?
```

```
    X-SAP-LogonToken={X-SAP-LogonToken}">
</img>
```

To do this, use the `imageUrl` parameter with an appropriate value in the `GET` call:

```
GET /documents/6406/reports/1?
reference=1.D.7&imageUrl=http%3A%2F%2FmyServer%2FmyPage.jsp
```

The resulting image link looks like:

```
<img border="0"
style="position:absolute;top:0px;left:0px;width:400px;height:300px;"
    alt="3D Column Chart" src="http://myServer/myPage.jsp?
documentId=6406&reportId=
    1&imageId=dxXMLDraft.drilloff_1*2*6&X-SAP-LogonToken={X-SAP-LogonToken}">
</img>
```

Response

Response type:

- `text/html` for HTML
- `application/zip` for zipped HTML
- `multipart/related` for MHTML
- `text/xml`
- `application/pdf`
- `application/vnd.ms-excel` for Microsoft Excel 2003
- `application/vnd.openxmlformats-officedocument.spreadsheetml.sheet` for Microsoft Excel 2007
- `text/csv` for CSV
- `text/plain` for plain text

Response body: the exported report in the specified format.

Example: HTML

```
curl -G -s -H "accept:text/html" -H X-SAP-LogonToken:"""%tokenValue%"""
 "<base_webi_REST_URL>/documents/9512/reports/67?chartOutputFormat=jpeg" >
exportedreport.htm
```

Example: Plain Text

```
GET /documents/7101/reports/1
```

```
Thread ID: 18
```

```
Accept: text/plain
Content-Type: text/plain
HTTP Response Code: 200
Response Time: 4709 ms
Response:
Report 1
Year    State    Lines    Sales revenue
2004    California    Accessories    $489,666
2004    California    City Skirts    $11,072
2004    California    City Trousers    $10,935
2004    California    Dresses    $110,210
...
```

Related Information

Getting the List of Documents [page 226]
Getting the Charsets [page 270]
Getting the List of Reports [page 341]

8.2.9.2 Exporting a Report as a Series of Pages

Usage

Exports a report of a Web Intelligence document as a series of pages in one of the following formats:

- XML
- PDF
- Microsoft Excel 2003
- Microsoft Excel 2007
- Text/JSON for JSON output

Request

`GET /documents/<documentID>/reports/<reportID>/pages?<optional_parameters>`

Optional Parameters

Parameter	Description	Supported Formats
`dpi`	Resolution in dots per inch (dpi) for generated charts. Value between 75 and 9600. Default is 300 for PDF format, 96 for all other formats.	All
`mode`	`normal` (default) or `quickDisplay`.	All

Parameter	Description	Supported Formats
`unit`	A string that defines the unit that sizes will be reported in. Values are `"metric"`, `"millimeter"`, `"point"`, and `"pixel"` (default).	XML
`rawValues`	Boolean. Default is `false`. If `true`, the raw values and their types are exported with the formatted values.	XML
`optimized`	Boolean. Default is `false`. If `true`, the generated output is optimized for calculations inside Microsoft Excel.	Microsoft Excel 2003 and Microsoft Excel 2007
`orientation`	Page orientation. Use to force a specific page orientation. Values are `portrait` and `landscape`.	If `mode=normal`
`widthScaling`	Number of pages per report displaying in width. The document setting applies by default.	If `mode=normal`
`heightScaling`	Number of pages per report displaying in height. The document setting applies by default.	If `mode=normal`

Response

Response type:

- `text/xml`
- `application/pdf`
- `application/vnd.ms-excel` for Microsoft Excel 2003
- `application/vnd.openxmlformats-officedocument.spreadsheetml.sheet` for Microsoft Excel 2007

Response body: the report in the expected format.

Example: XML

```
curl -G -s -H "accept:text/xml" -H X-SAP-LogonToken:"""<tokenValue>"""
 "<base_webi_REST_URL>/documents/9227/reports/1/pages" > xmlreport.xml
```

Example: PDF

```
curl -G -s -H "accept:application/pdf" -H X-SAP-LogonToken:"""<tokenValue>"""
 "<base_webi_REST_URL>/documents/9227/reports/1/pages?
mode=normal&widthScaling=1" > pdfreport.pdf
```

Example: Microsoft Excel 2003

```
curl -G -s -H "accept:application/vnd.ms-excel" -H "X-SAP-
LogonToken:""<tokenValue>"""
 "<base_webi_REST_URL>/documents/9227/reports/1/pages" > excel2003.xls
```

Example: Microsoft Excel 2007

```
curl -G -s -H "accept:application/vnd.openxmlformats-
officedocument.spreadsheetml.sheet"
 -H X-SAP-LogonToken:"""<tokenValue>""" "<base_webi_REST_URL>/documents/9227/
reports/1/pages" > excel2007.xlsx
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.2.9.3 Exporting a Page or Range of Pages of a Report

Usage

Exports one given page or a range of pages of a report in one of the following formats:

- HTML
- Zipped HTML
- MHTML (multipart HTML)
- XML
- PDF
- Microsoft Excel 2003
- Microsoft Excel 2007

> **Note**
>
> - If you choose the HTML format, the web service generates the image links. Therefore, the logon token must still be valid when the HTML output is displayed.
> - Exports to HTML will be optimized for your browser if you use the User-Agent HTTP header in the REST API call.

Request

```
GET /documents/<documentID>/reports/<reportID>/pages/<pageRange>?
<optional_parameters>
```

Where:

- `<pageRange>` is a pages range defined by the following syntax:
  Where `<pageRange>`:
  - `<pageRange>` = {subPageRange}[,{subPageRange}[,{subPageRange}...]]
  - `<subPageRange>` = {pageSequence} or {simplePage} or {slot} or {iRef}
  - `<pageSequence>` = {simplePage1}-{simplePage2} → pages simplePage1 to simplePage2 of the report
  - `<simplePage>` = {pageNumber} or "last&quot" for the last page. The first page is 1
  - `<slot>` = {x}:{y} → slot x:y of report (lower value for both x and y is 1)
  - `<iRef>`

> **Note**
>
> Since SAP BI 4.2 SP4, you can use a page range instead of a page number.

Example: `GET /documents/<7610>/reports/<32>/pages/9-15,2:1,last,X.R.8.1,3`

> **Note**
>
> `<compositePageRange>` (i.e. combining several subPageRanges with a comma) and pageSequence are not allowed for media types which do not support multiple pages output (HTML, MHTML, ZIP). An error will then be thrown.
>
> ```
> HTTP Response Code: 400
> Response body:
> <?xml version="1.0" encoding="UTF-8"?>
> <error>
>  <error_code>WSR 00100</error_code>
>  <message>Rule not respected ("This media type does not support multiple
> pages output.")</message>
> </error>
> ```

> **→ Remember**
>
> To export the last page of the report, use:
>
> `GET /documents/<documentID>/reports/<reportID>/pages/last`

Optional Parameters

Parameter	Description	Supported Formats
`dpi`	Resolution in dots per inch (dpi) for generated charts. Value between 75 and 9600. Default is 300 for PDF format, 96 for all other formats.	All
`mode`	`normal` (default) or `quickDisplay`.	All

Parameter	Description	Supported Formats
`orientation`	Page orientation. Used to force a specific page orientation. Values are `portrait` and `landscape`.	If `mode=normal`
`widthScaling`	Number of pages per report displaying in width. The document setting applies by default.	If `mode=normal`
`heightScaling`	Number of pages per report displaying in height. The document setting applies by default.	If`mode=normal`
`chartOutputFormat`	Output format for generated chart. Values are: `jpeg`, `bmp`, `gif`, and `png` (default).	HTML, ZIP and MHTML
`imageUrl`	String used to customize image links when they cannot be reached from the information system.	HTML
`fileName`	A string that defines the name of the ZIP file. If set, the HTML output is zipped to a `fileName.zip` file. Maximum string length is 96.	ZIP
`unit`	A string that defines the unit that sizes will be reported in. Values are `"metric"`, `"millimeter"`, `"point"`, and `"pixel"` (default).	XML
`rawValues`	Boolean. Default is `false`. If `true`, the raw values and their types are exported with the formatted values.	XML
`optimized`	Boolean. Default is `false`. If `true`, the generated output is optimized for calculations inside Microsoft Excel..	Microsoft Excel 2003 and Microsoft Excel 2007

Customization of Image Links

You may need to customize the source of an image in a page exported as HTML if the image is unreachable by your system:

```
<img border="0"
style="position:absolute;top:0px;left:0px;width:400px;height:300px;"
    alt="3D Column Chart" src="http://dewdfwadept1054.dhcp.wdf.sap.corp:6405/
biprws/
    raylight/v1/documents/6406/reports/1/pages/1/images/
dxXMLDraft.drilloff_1*2*6?
    X-SAP-LogonToken={X-SAP-LogonToken}">
</img>
```

To do this, use the `imageUrl` parameter with an appropriate value in the `GET` call:

```
GET /documents/6406/reports/1/pages/0?
reference=1.D.7&imageUrl=http%3A%2F%2FmyServer%2FmyPage.jsp
```

The resulting image link looks like:

```
<img border="0"
style="position:absolute;top:0px;left:0px;width:400px;height:300px;"
    alt="3D Column Chart" src="http://myServer/myPage.jsp?
documentId=6406&reportId=
    1&pageId=1&imageId=dxXMLDraft.drilloff_1*2*6&X-SAP-LogonToken={X-SAP-
LogonToken}">
</img>
```

Response

Response type:

- `text/html`
- `application/zip` for zipped HTML
- `multi-part/related` for MHTML
- `text/xml`
- `application/pdf`
- `application/vnd.ms-excel` for Microsoft Excel 2003
- `application/vnd.openxmlformats-officedocument.spreadsheetml.sheet` for Microsoft Excel 2007

Response body: the report in the expected format.

Example: XML

```
curl -H "accept:text/xml" -H X-SAP-LogonToken:"""%tokenValue%"""
 "<base_webi_REST_URL>/documents/9646/reports/479/pages/1" > page1.xml
```

Example: PDF

```
curl -H "accept:application/pdf" -H X-SAP-LogonToken:"""<tokenValue>"""
 "<base_webi_REST_URL>/documents/9646/reports/479/pages/2" > page2.pdf
```

Example: Microsoft Excel 2003

```
curl -H "accept:application/vnd.ms-excel" -H X-SAP-LogonToken:"""<tokenValue>"""
 "<base_webi_REST_URL>/documents/9646/reports/479/pages/3" > page3.xls
```

Example: Microsoft Excel 2007

```
curl -G -s -H "accept:application/vnd.openxmlformats-
officedocument.spreadsheetml.sheet"
 -H X-SAP-LogonToken:"""<tokenValue>""" "<base_webi_REST_URL>/documents/9646/
reports/479/pages/1"
 > page1.xlsx
```

8.2.10 Shared Element APIs

The REST APIs to manage shared elements in the CMS repository, documents, or reports.

The tables below indicate the 4.2 release in which support for the API was introduced.

Base URL: `http://<server_name>:6405/biprws/raylight/v1`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<reportID>`: report identifier retrieved from the list of reports
- `<elementID>`: report element identifier retrieved from the list of report elements
- `<sharedelementID>`: shared element identifier retrieved from the list of shared elements

Query parameters:

- `<sharedelementId>`: shared element identifier retrieved from the list of shared elements

See also Managing Shared Elements [page 679]

Shared Elements in the CMS Repository

Action	Method	URL	Since
Getting the List of Shared Elements [page 679]	GET	/sharedelements	4.2 SP3
Getting the Details of a Shared Element [page 681]	GET	/sharedelements/<sharedelementID>	4.2 SP3
Editing a Shared Element [page 683]	PUT	/sharedelements/<sharedelementID>	4.2 SP3
Deleting a Shared Element [page 684]	DELETE	/sharedelements/<sharedelementID>	4.2 SP3

Shared Elements of a Document

Action	Method	URL	Since
Getting the List of Shared Elements in a Document [page 685]	GET	/documents/<documentID>/sharedelements	4.2 SP3
Updating all Shared Elements of a Document [page 687]	PUT	/documents/<documentID>/sharedelements	4.2 SP3

Action	Method	URL	Since
Updating a Shared Element of a Document [page 686]	PUT	/documents/<documentID>/sharedelements/<sharedelementID>	4.2 SP3

Shared Elements Linked to Report Elements in a Document

Action	Method	URL	Since
Creating a Report Element From a Shared Element [page 391]	POST	/documents/<documentID>/reports/<reportID>/elements?sharedelementId=<sharedelementId>	4.2 SP3
Saving a Report Element as a Shared Element [page 688]	POST	/documents/<documentID>/reports/<reportID>/elements/<elementID>/sharedelement	4.2 SP3
Getting the Details of a Shared Element Linked to a Report Element [page 689]	GET	/documents/<documentID>/reports/<reportID>/elements/<elementID>/sharedelement	4.2 SP3
Unlink a Shared Element From a Report Element [page 690]	DELETE	/documents/<documentID>/reports/<reportID>/elements/<elementID>/sharedelement	4.2 SP3

8.2.11 Updating the Structure of a Report

> **Note**
>
> Prefer using the report element APIs to get Web Intelligence report content. See Managing Report Elements [page 390].

Usage

Updates the report structure of the specified report.

Request

`PUT /documents/<documentID>/reports/<reportID>/specification`

Request type: `text/xml`

Request body: the report structure

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: To Add a Cell to a Report Structure

```
PUT /documents/2334/reports/23/specification
```

Request body:

```
<REPORT rId="12" name="My First Cell">
    <PAGE_BODY>
        <CELL  x="1000" y="1500">
            <CONTENT>="My First CELL"</CONTENT>
        </CELL>
    </PAGE_BODY>
</REPORT>
```

Response:

```
<success>
    <message>The resource of type "Report" with  identifier "23" has been
successfully updated.</message>
    <id>23</id>
</success>
```

Example: To Add a Vertical Table to a Report Structure

Request body:

```
<REPORT name="Vertical Table">
    <PAGE_HEADER/>
    <PAGE_BODY>
        <VTABLE name="My Vertical Table" x="3037" y="4455">
            <ROWGROUP type="HEADER">
                <TR>
                    <TDCELL>
                        <CONTENT>=NameOf([Country])</CONTENT>
                    </TDCELL>
                    <TDCELL>
                        <CONTENT>=NameOf([Revenue])</CONTENT>
                    </TDCELL>
                </TR>
            </ROWGROUP>
            <ROWGROUP type="BODY">
                <TR>
                    <TDCELL>
                        <CONTENT>=[Country]</CONTENT>
                    </TDCELL>
                    <TDCELL>
                        <CONTENT>=[Revenue]</CONTENT>
                    </TDCELL>
                </TR>
```

```
            </ROWGROUP>
            <ROWGROUP type="FOOTER">
                <TR>
                    <TDCELL>
                        <CONTENT/>
                    </TDCELL>
                    <TDCELL>
                        <CONTENT>=[Revenue]</CONTENT>
                    </TDCELL>
                </TR>
            </ROWGROUP>
        </VTABLE>
    </PAGE_BODY>
    <PAGE_FOOTER/>
</REPORT>
```

Example: To Add a Vertical Table with Break

Request body:

```
<REPORT name="Vertical Table And Break">
    <PAGE_HEADER/>
    <PAGE_BODY>
        <VTABLE name="Table And Break " x="3037" y="4455">
            <AXIS>
                <EXPRS>
                    <AXIS_EXPR>=[Country]</AXIS_EXPR>
                    <AXIS_EXPR>=[Resort]</AXIS_EXPR>
                </EXPRS>
                <BREAK bId="1" expr="=[Country]" addSort="yes" onePage="yes"
newPage="yes" duplicate="center"/>
            </AXIS>
            <ROWGROUP breakId="1" type="HEADER">
                <TR>
                    <TDCELL>
                        <CONTENT>=NameOf([Country])</CONTENT>
                    </TDCELL>
                    <TDCELL>
                        <CONTENT>=NameOf([Resort])</CONTENT>
                    </TDCELL>
                    <TDCELL>
                        <CONTENT>=NameOf([Revenue])</CONTENT>
                    </TDCELL>
                </TR>
            </ROWGROUP>
            <ROWGROUP type="BODY">
                <TR>
                    <TDCELL>
                        <STYLE>
                            <TEXTVALIGN value="center"/>
                        </STYLE>
                        <CONTENT>=[Country]</CONTENT>
                    </TDCELL>
                    <TDCELL>
                        <CONTENT>=[Resort]</CONTENT>
                    </TDCELL>
                    <TDCELL>
                        <CONTENT>=[Revenue]</CONTENT>
                    </TDCELL>
                </TR>
            </ROWGROUP>
            <ROWGROUP breakId="1" type="FOOTER">
                <TR>
```

```
                    <TDCELL>
                        <CONTENT>=[Country]</CONTENT>
                    </TDCELL>
                    <TDCELL><CONTENT/></TDCELL>
                    <TDCELL><CONTENT/></TDCELL>
                </TR>
            </ROWGROUP>
        </VTABLE>
    </PAGE_BODY>
    <PAGE_FOOTER/>
</REPORT>
```

Example: To Add a Vertical Table with one Filter on the Report and one Filter on the Table

Request body:

```
<REPORT name="Filter">
    <DATA>
        <DATA_FILTER>
            <WHERE>
                <FILTER key="[Country]">
                    <CONDITION operatorCondition="InList">
                        <MEMBER>France</MEMBER>
                    </CONDITION>
                </FILTER>
            </WHERE>
        </DATA_FILTER>
    </DATA>
    <PAGE_BODY>
        <VTABLE bId="17" name="Block 1" x="3749" y="3443">
            <AXIS>
                <EXPRS>
                    <AXIS_EXPR>=[Country]</AXIS_EXPR>
                    <AXIS_EXPR>=[Resort]</AXIS_EXPR>
                </EXPRS>
            </AXIS>
            <DATA>
                <DATA_FILTER>
                    <WHERE>
                        <FILTER key="[Resort]">
                            <CONDITION operatorCondition="InList">
                                <MEMBER>French Riviera</MEMBER>
                            </CONDITION>
                        </FILTER>
                    </WHERE>
                </DATA_FILTER>
            </DATA>
            <ROWGROUP type="HEADER">
                <TR height="567">
                    <TDCELL>
                        <CONTENT>=NameOf([Country])</CONTENT>
                    </TDCELL>
                    <TDCELL>
                        <CONTENT>=NameOf([Resort])</CONTENT>
                    </TDCELL>
                    <TDCELL>
                        <CONTENT>=NameOf([Revenue])</CONTENT>
                    </TDCELL>
                </TR>
            </ROWGROUP>
            <ROWGROUP type="BODY">
```

```
                <TR height="567">
                    <TDCELL>
                        <CONTENT>=[Country]</CONTENT>
                    </TDCELL>
                    <TDCELL>
                        <CONTENT>=[Resort]</CONTENT>
                    </TDCELL>
                    <TDCELL>
                        <CONTENT>=[Revenue]</CONTENT>
                    </TDCELL>
                </TR>
            </ROWGROUP>
        </VTABLE>
    </PAGE_BODY>
</REPORT>
```

Example: To Add a Section with a Table and a Cell

Request body:

```
<REPORT name="Section &amp; Filter">
    <DATA/>
    <PAGE_BODY>
        <ROW height="6255"/>
        <SECTION>
            <DATA>
                <DATA_FILTER>
                    <WHERE>
                        <FILTER key="[Country]">
                            <CONDITION operatorCondition="InList">
                                <MEMBER>France</MEMBER>
                            </CONDITION>
                        </FILTER>
                    </WHERE>
                </DATA_FILTER>
            </DATA>
            <AXIS>
                <SORTS>
                    <SORT sign="desc" expr="=[Country]"/>
                </SORTS>
                <AXIS_EXPR>=[Country]</AXIS_EXPR>
            </AXIS>
            <SBODY bottomPadding="3638" bookmark="yes">
                <CELL class="ia-section-cell" x="600" y="500">
                    <CONTENT>=[Country]</CONTENT>
                </CELL>
                <VTABLE name="Block 1" x="3749" y="500">
                    <DATA/>
                    <ROWGROUP type="HEADER">
                        <TR>
                            <TDCELL>
                                <CONTENT>=NameOf([Resort])</CONTENT>
                            </TDCELL>
                            <TDCELL>
                                <CONTENT>=NameOf([Revenue])</CONTENT>
                            </TDCELL>
                        </TR>
                    </ROWGROUP>
                    <ROWGROUP type="BODY">
                        <TR>
                            <TDCELL>
                                <CONTENT>=[Resort]</CONTENT>
                            </TDCELL>
```

```
                                <TDCELL>
                                    <CONTENT>=[Revenue]</CONTENT>
                                </TDCELL>
                            </TR>
                        </ROWGROUP>
                    </VTABLE>
                </SBODY>
            </SECTION>
            <ROW height="1500"/>
        </PAGE_BODY>
</REPORT>
```

Example: To Add Cells with hideAlways Attribute

Request body:

```
<REPORT name="Cell with hide always and Cell with hide conditional">
    <DATA/>
    <PAGE_BODY>
        <ROW height="2992" bottomPadding="0">
            <CELL x="7416" y="432" hideAlways="yes" width="15048" height="1547">
                <CONTENT>ALWAYS HIDE</CONTENT>
            </CELL>
            <CELL x="7416" y="2000" hideAlways="no"
hideExpr="=Count([Country])>3"  width="15048" height="1547">
                <CONTENT>=Count([Country])>3</CONTENT>
            </CELL>
        </ROW>
    </PAGE_BODY>
</REPORT>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.2.12 Managing Data Filters in Reports

A series of APIs is provided to perform tasks on the data filters of a report.

Creating a Data Filter [page 366]

Getting the Details of a Data Filter [page 367]

Updating a Data Filter [page 368]

Deleting a Data Filter [page 370]

8.2.12.1 Creating a Data Filter

Usage

Creates a data filter for a specific report, referenced by its ID.

Request

`POST /documents/<documentID>/reports/<reportID>/datafilter`

Request body: the description of the data filter. The `key` attribute should be one of the `formulaLanguageId` elements of the report.

The operator is one of the available report filters :

- `Equal`
- `NotEqual`
- `Greater`
- `GreaterOrEqual`
- `Less`
- `LessOrEqual`
- `Between`
- `NotBetween`
- `InList`
- `NotInList`
- `IsNull`
- `IsNotNull`
- `IsAny`
- `Like`
- `NotLike`
- `Both`
- `Except`

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
POST /documents/127/reports/1/datafilter
```

Request body:

```
<datafilter>
    <and>
        <condition key="[Lines]" operator="InList">
            <value>City Trousers</value>
            <value>Shirt Waist</value>
            <value>Jackets</value>
        </condition>
        <condition key="[State]" operator="NotEqual">
            <value>Colorado</value>
        </condition>
        <or>
            <condition key="[Sales revenue]" operator="Greater">
                <value>5000</value>
            </condition>
            <condition key="[Sales revenue]" operator="Less">
                <value>10</value>
            </condition>
        </or>
    </and>
</datafilter>
```

Response:

```
<success>
    <message>The resource of type "Report" with identifier "1" has been
successfully updated.</message>
    <id>1</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.2.12.2 Getting the Details of a Data Filter

Usage

Gets the description of the data filter of a specific report, referenced by its ID.

Request

```
GET /documents/<documentID>/reports/<reportID>/datafilter
```

Response

Response type: `application/xml` or `application/json`

Response body: the description of the data filter. The `key` attribute should be one of the `formulaLanguageId` elements of the report.

Example

```
GET /documents/4567/reports/1/datafilter
```

Response:

```
<datafilter>
    <and>
        <condition key="[Lines]" operator="InList">
            <value>City Trousers</value>
            <value>Shirt Waist</value>
            <value>Jackets</value>
        </condition>
        <condition key="[State]" operator="NotEqual">
            <value>Colorado</value>
        </condition>
        <or>
            <condition key="[Sales revenue]" operator="Greater">
                <value>5000</value>
            </condition>
            <condition key="[Sales revenue]" operator="Less">
                <value>10</value>
            </condition>
        </or>
    </and>
</datafilter>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.2.12.3 Updating a Data Filter

Usage

Updates the data filter applied to a given report specified in the URL parameter.

Request

`PUT /documents/<documentID>/reports/<reportID>/datafilter`

Request body: the description of the data filter to update. The operator is one of the available report filter conditions:

- `Equal`
- `NotEqual`
- `Greater`
- `GreaterOrEqual`
- `Less`
- `LessOrEqual`
- `Between`
- `NotBetween`
- `InList`
- `NotInList`
- `IsNull`
- `IsNotNull`
- `IsAny`
- `Like`
- `NotLike`
- `Both`
- `Except`

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`PUT /documents/223/reports/1/datafilter`

Request body:

```
<datafilter>
    <and>
        <condition key="[Lines]" operator="InList">
            <value>City Trousers</value>
            <value>Shirt Waist</value>
            <value>Jackets</value>
        </condition>
        <condition key="[State]" operator="NotEqual">
            <value>Colorado</value>
```

```
        </condition>
        <or>
            <condition key="[Sales revenue]" operator="Greater">
                <value>5000</value>
            </condition>
            <condition key="[Sales revenue]" operator="Less">
                <value>10</value>
            </condition>
        </or>
    </and>
</datafilter>
```

Response:

```
<success>
    <message>The resource of type "Report" with identifier "1" has been
successfully updated.</message>
    <id>1</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.2.12.4 Deleting a Data Filter

Usage

Deletes a data filter from a specific report, referenced by its ID.

Request

```
DELETE /documents/<documentID>/reports/<reportID>/datafilter
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/9512/reports/67/datafilter
```

Response:

```
<success>
    <message>The resource of type "Report" with identifier "67" has been
successfully updated.</message>
    <id>67</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.2.13 Getting the Structure of a Report

> **Note**
>
> Prefer using the report element APIs to get Web Intelligence report content. See Managing Report Elements [page 390].

Usage

Gets the structure of a report.

Request

```
GET /documents/<documentID>/reports/<reportID>/specification
```

Response

Response type: `text/xml`

Response body: the description of the report structure.

Example

```
GET /documents/2334/reports/23/specification
```

Response:

```
<REPORT rId="12" name="Report1">
    <PAGE_HEADER bId="1"/>
    <PAGE_BODY bId="2">
        <VTABLE y="100" x="100" bId="21"name="Table 1">
           <ROWGROUP type="header">
                <TR>
                    <TDCELL bId="211"><CONTENT>=nameof([YEAR])</CONTENT></TDCELL>
                    <TDCELL bId="212"><CONTENT>=nameof([INCOME])</CONTENT></
TDCELL>
                </TR>
            </ROWGROUP>
            <ROWGROUP type="body">
                <TR>
                    <TDCELL bId="213"><CONTENT>=[YEAR]</CONTENT></TDCELL>
                    <TDCELL bId="214"><CONTENT>=[INCOME]</CONTENT></TDCELL>
                </TR>
            </ROWGROUP>
        </VTABLE>
    </PAGE_BODY>
    <PAGE_FOOTER bId="3"/>
</REPORT>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.2.14 Managing Report Comments

You can visible comments for a report or a range of pages in a report.

Related Information

Managing Reports [page 339]
Getting the Visible Comments of a Page Range of a Report [page 374]

8.2.14.1 Getting the Visible Comments of a Report

Usage

This call returns the visible comments of a report of a Web Intelligence document. This involves exporting the specified report in paginated mode, and returns only the comments of the generated output.

Request

```
GET /documents/<documentID>/reports/<reportID>/pages/comments
```

Parameters

Required

- `<documentID>`: (type=integer) a valid identifier of a Web Intelligence document.
- `<reportID>`: (type=integer) a valid identifier of a report inside the Web Intelligence document.

Optional

- `<mode>`: (type=string, values="normal" or "quickDisplay", default="normal")
- `<orientation>`: (type=string, values="portrait" or "landscape") force page orientation if specified.
- `<widthScaling>`: (type=integer, min=0, default=the settings in the original document) number of pages per report displaying in width, 0 meaning no constraint in width.
- `<heightScaling>`: (type=integer, min=0, default=the settings in the original document) number of pages per report displaying in height, 0 meaning no constraint in height.

Response

Response type: `application/xml` or `application/json`

Example: XML

```
[GET] <url>/documents/22684/reports/19/pages/comments
```

Response:

```
<comments>
    <comment>
        <id>80</id>
        <created>2016-08-16T15:18:03.000Z</created>
```

```
        <updated>2016-08-16T15:18:03.000Z</updated>
        <createdBy>
            <name>raylight_user</name>
            <cuid>AZFmWPGBwwRBm3almUJGkfk</cuid>
        </createdBy>
        <lastAuthor>
            <name>raylight_user</name>
            <cuid>AZFmWPGBwwRBm3almUJGkfk</cuid>
        </lastAuthor>
        <value>Comments from the report are here.</value>
        <element>
            <id>4</id>
            <reference>J.4</reference>
        </element>
    </comment>
    ...
</comments>
```

8.2.14.2 Getting the Visible Comments of a Page Range of a Report

Usage

This call returns the visible comments of a page range of a report of a Web Intelligence document. This involves exporting the specified page range and returns only the comments of the generated output

Request

```
GET /documents/<documentID>/reports/<reportID>/pages/<pageRange>/comments
```

Parameters

Required

- `<documentID>`: (type=integer) a valid identifier of a Web Intelligence document.
- `<reportID>`: (type=integer) a valid identifier of a report inside the Web Intelligence document.
- `<pageRange>`: (type=integer) a valid page range. See About report page ranges [page 375].

Optional

- `<mode>`: (type=string, values="normal" or "quickDisplay", default="normal")
- `<orientation>`: (type=string, values="portrait" or "landscape") force page orientation.
- `<widthScaling>`: (type=integer, min=0, default=the settings in the original document) number of pages per report displaying in width, 0 meaning no constraint in width.
- `<heightScaling>`: (type=integer, min=0, default=the settings in the original document) number of pages per report displaying in height, 0 meaning no constraint in height.

Response

Response type: `application/xml` or `application/json`

Example: XML

```
[GET] <url>/documents/22684/reports/19/pages/1/comments
```

Response:

```
<comments>
    <comment>
        <id>80</id>
        <created>2016-08-16T15:18:03.000Z</created>
        <updated>2016-08-16T15:18:03.000Z</updated>
        <createdBy>
            <name>raylight_user</name>
            <cuid>AZFmWPGBwwRBm3almUJGkfk</cuid>
        </createdBy>
        <lastAuthor>
            <name>raylight_user</name>
            <cuid>AZFmWPGBwwRBm3almUJGkfk</cuid>
        </lastAuthor>
        <value>Comments in the document.</value>
        <element>
            <id>4</id>
            <reference>J.4</reference>
        </element>
    </comment>
    ...
</comments>
```

8.2.14.2.1 About report page ranges

Since 4.2 SP04, `pageIndex` has been superseded by the more versatile `pageRange` parameter.

Parameter values

- `pageRange` = {subPageRange}[,{subPageRange}[,{subPageRange}...]]
- `subPageRange` = {pageSequence} or {simplePage} or {slot} or {iRef}
- `pageSequence` = {simplePage1}-{simplePage2} → pages simplePage1 to simplePage2 of the report
- `simplePage` = {pageNumber} or "last" → first page is still 1
- `slot` = {x}:{y} → slot x:y of report (lower value for both x and y is 1) (see attached page for an explanation on slots)
- `iRef` = the reference of (an instance of) a report element. (since 4.2 SP4)

Example

```
[GET] /documents/7610/reports/32/pages/9-15,2:1,last,X.R.8.1,3
```

> **Note**
>
> `CompositePageRange` (i.e. combining several `subPageRanges` with a comma) and pageSequence are not allowed for media types which do not support multiple pages output (HTML, MHTML, ZIP). An error will then be thrown:
>
> ```
> HTTP Response Code: 400
> Response body:
> <?xml version="1.0" encoding="UTF-8"?>
> <error>
>  <error_code>WSR 00100</error_code>
>  <message>Rule not respected ("This media type does not support multiple
> pages output.")</message>
> </error>
> ```

8.3 Drilling on Report Data

You use the drill function to analyze the results displayed in reports. Drilling on reports lets you look deeper into your data to discover the details behind a summary result displayed in tables, charts, or sections.

Getting the Drill Mode [page 377]

Updating the Drill Mode [page 377]

Enabling the Query Drill [page 379]

Disabling the Query Drill [page 380]

Getting Information on the Drill Hierarchies [page 380]

Getting the Free Drill Elements [page 382]

Creating a Drill Filter [page 383]

Getting the Drill Filters of a Report [page 384]

Getting the Details of a Drill Filter [page 385]

Updating a Drill Filter [page 386]

Removing a Drill Filter [page 387]

Performing a Drill [page 388]

Making a Snapshot of a Report in Drill Mode [page 390]

8.3.1 Getting the Drill Mode

Usage

Obtains the current drill information.

Returns an error if the drill is not activated on the report.

Request

```
GET /documents/<documentID>/reports/<reportID>/driller
```

Response

Response type: `application/xml` or `application/json`

Response body: the drill settings, where `<output>` specifies the drill mode (`true`|`false`).

Example

```
GET /documents/4567/reports/1/driller
```

Response:

```
<driller>
    <output>true</output>
</driller>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.3.2 Updating the Drill Mode

Usage

Updates the drill mode of a report.

Request

`PUT /documents/<documentID>/reports/<reportID>/driller`

Request body:

```
<driller>
    <output>
```

`<output>` specifies the drill mode (`true|false`).

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`PUT /documents/3422/reports/2/driller`

Request body:

```
<driller>
    <output>false</output>
</driller>
```

Response:

```
<success>
    <message>The resource of type "Driller" has been successfully updated.</
message>
    <id>1</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.3.3 Enabling the Query Drill

Usage

Activates the drill on a report.

Request

`POST /documents/<documentID>/reports/<reportID>/driller`

Request body (optional):

```
<driller>
    <output>
```

`<output>` specifies the drill mode (`true|false`). If not provided, the drill is activated with the default settings (`true`).

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`POST /documents/4567/reports/1/driller`

Response:

```
<success>
    <message>The resource of type "Driller" has been successfully created.</
message>
    <id>2</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.3.4 Disabling the Query Drill

Usage

Disables the drill on the specified report.

Request

```
DELETE /documents/<documentID>/reports/<reportID>/driller
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /8022/reports/1/driller
```

Response:

```
<success>
    <message>The resource of type "Driller" has been successfully removed.</
message>
    <id>2</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.3.5 Getting Information on the Drill Hierarchies

Usage

Obtains the drill hierarchies.

It returns an error if the drill is not activated on this report.

Request

```
GET /documents/<documentID>/reports/<reportID>/driller/hierarchies
```

Response

Response type: `application/xml` or `application/json`

Response body: the details of the drill hierarchies.

Example

```
GET /documents/4567/reports/1/driller/hierarchies
```

Response:

```
<hierarchies>
    <hierarchy>
        <id>DP0.DH1</id>
        <name>Resort Hierarchy</name>
        <dataProviderId>DP0</dataProviderId>
        <elements>
            <element>
                <id>DP0.DO39</id>
                <name>Country</name>
                <description></description>
                <filterValue></filterValue>
                <qualification>dimension</qualification>
                <inQuery>false</inQuery>
                <ambiguous>false</ambiguous>
                <inScope>true</inScope>
            </element>
            <element>
                <id>DP0.DO2</id>
                <name>Resort</name>
                <description></description>
                <filterValue></filterValue>
                <qualification>dimension</qualification>
                <inQuery>false</inQuery>
                <ambiguous>false</ambiguous>
                <inScope>true</inScope>
            </element>
            <element>
                <id>DP0.DO4</id>
                <name>Service Line</name>
                <description></description>
                <filterValue></filterValue>
                <qualification>dimension</qualification>
                <inQuery>false</inQuery>
                <ambiguous>false</ambiguous>
                <inScope>true</inScope>
```

```
            </element>
            <element>
                <id>DP0.DO5</id>
                <name>Service</name>
                <description></description>
                <filterValue></filterValue>
                <qualification>dimension</qualification>
                <inQuery>false</inQuery>
                <ambiguous>false</ambiguous>
                <inScope>true</inScope>
            </element>
        </elements>
    </hierarchy>
</hierarchies>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.3.6 Getting the Free Drill Elements

Usage

Obtains the free drill elements.

Returns an error if the drill is not activated on the report.

Request

```
GET /documents/<documentID>/reports/<reportID>/driller/drillelements
```

Response

Response type: `application/xml` or `application/json`

Response body: the details of the available drill elements of the report:

- `qualification`
- `<id>`
- `<name>`

Example

```
GET /documents/4567/reports/1/driller/drillelements
```

Response:

```
<drillelements>
    <drillelement qualification="Dimension" inQuery="false" ambiguous="false"
inScope="true">
        <id>DP0.DO39</id>
        <name>Country</name>
    </drillelement>
    <drillelement qualification="Dimension" inQuery="false" ambiguous="false"
inScope="true">
        <id>DP0.DO2</id>
        <name>Resort</name>
    </drillelement>
    <drillelement qualification="Dimension" inQuery="false" ambiguous="false"
inScope="true">
        <id>DP0.DO4</id>
        <name>Service Line</name>
    </drillelement>
    <drillelement qualification="Dimension" inQuery="false" ambiguous="false"
inScope="true">
        <id>DP0.DO5</id>
        <name>Service</name>
    </drillelement>
</drillelements>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.3.7 Creating a Drill Filter

Usage

Creates a drill filter for a report.

Returns an error if the drill is not activated on the report.

Request

```
POST /documents/<documentID>/reports/<reportID>/driller/filters
```

Request body: the identifier and value of the report drill filter.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
POST /documents/127/reports/1/driller/filters
```

Request body:

```
<filter>
    <id>DP1.DO22</id>
    <value>France</value>
</filter>
```

Response:

```
<success>
    <message>The resource of type "DrillFilter" with identifier "DP1.DO22" has
been successfully created.</message>
    <id>DP1.DO22</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.3.8 Getting the Drill Filters of a Report

Usage

Get the list of drill filters for a report.

Returns an error if the drill is not activated on the report.

Request

```
GET /documents/<documentID>/reports/<reportID>/driller/filters
```

Response

Response type: `application/xml` or `application/json`

Response body: the description of the available filters for the report.

Example

```
GET /documents/4567/reports/1/driller/filters
```

Response:

```
<filters>
    <filter qualification="Dimension" inQuery="false" ambiguous="false"
inScope="true">
        <id>DP0.DO39</id>
        <name>Country</name>
    </filter>
</filters>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.3.9 Getting the Details of a Drill Filter

Usage

Gets the details of a drill filter for a report.

Returns an error if the drill is not activated on the report.

Request

```
GET /documents/<documentID>/reports/<reportID>/driller/filters/<filterID>
```

Response

Response type: `application/xml` or `application/json`

Response body: the description of the drill filter.

Example

```
GET /documents/13343/reports/1/driller/filters/DP0.DO13
```

Response:

```
<filter qualification="Dimension" inQuery="false" ambiguous="false"
inScope="true">
    <id>DP0.DO13</id>
    <name>City</name>
    <lov hierarchical="false" partial="false" refreshable="false">
        <values>
            <value>Albertville</value>
            <value>Augsburg</value>
            <value>Belfast</value>
            . . .
            <value>Washington D.C.</value>
            <value>Yokohama</value>
        </values>
        <columns mappingId="0">
            <column id="0" type="String"></column>
        </columns>
    </lov>
</filter>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the Drill Filters of a Report [page 384]

8.3.10 Updating a Drill Filter

Usage

Updates the definition of a drill filter for a report.

Returns an error if the drill is not activated on the report.

Request

```
PUT /documents/<documentID>/reports/<reportID>/driller/filters/<filterID>
```

Request body: the drill filter to update.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/223/reports/2/driller/filters/DP2.O11
```

Request body:

```
<filter>
    <value>Germany</value>
</filter>
```

Response:

```
<success>
    <message>The resource of type "DrillFilter" with identifier "DP2.O11" has
been successfully updated.</message>
    <id>DP2.O11</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the Drill Filters of a Report [page 384]

8.3.11 Removing a Drill Filter

Usage

Removes a drill filter for a report.

Returns an error if the drill is not activated on the report.

Request

```
DELETE /documents/<documentID>/reports/<reportID>/driller/filters/<filterID>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/9512/reports/67/driller/filters/DP5.O12
```

Response:

```
<success>
    <message>The resource of type "DrillFilter" with identifier "DP5.O12" has
been successfully deleted.</message>
    <id>DP5.012</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the Drill Filters of a Report [page 384]

8.3.12 Performing a Drill

Usage

Performs a drill on a report.

This throws an error if the resource is not created or if the parameters are not valid.

Request

```
POST /documents/<documentID>/reports/<reportID>/driller/instructions
```

Request body: the drill instructions. The drill instructions are made of the following:

- The instruction type (either `"Down"`, `"Up"`, or `"By"`)
- The identifier of the report element on which you perform the drill
- The drill elements on which you perform the drill

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
POST /documents/9512/reports/67/driller/instructions
```

Request body:

```
<instruction type="Down">
    <elementId>11</elementId>
    <from>
        <drillElement>
            <id>DP0.DO2</id>
            <filterValue>French Riviera</filterValue>
        </drillElement>
    </from>
    <to>
        <drillElement>
            <id>DP0.DO4</id>
            <hierarchyId>DP0.DH1</hierarchyId>
        </drillElement>
    </to>
</instruction>
```

Response:

```
<success>
    <message>The resource of type "DrillerInstruction" has been successfully
created.</message>
    <id>1</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.3.13 Making a Snapshot of a Report in Drill Mode

Usage

Makes a snapshot of a report in drill mode.

This throws an error if the resource is not created or if the parameters are not valid.

Request

```
POST /documents/<documentID>/reports/<reportID>/driller/snapshot
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
POST /documents/7738/reports/12/driller/snapshot
```

Response:

```
<success>
    <message>Resource of type "Report" with identifier "12" has been
successfully created.</message>
    <id>12</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.4 Managing Report Elements

Creating a Report Element [page 391]

Getting the List of Report Elements [page 396]
Getting the Details of a Report Element [page 400]
Getting the Background of a Report Element [page 411]
Getting the Datapaths of a Report Element [page 412]
Getting the Dataset of a Report Element [page 416]
Getting the Details of a Hierarchy Member [page 422]
Updating a Report Element [page 423]
Inserting a Row or Column into a Table [page 429]
Merging or Splitting Cells of a Table [page 430]
Setting a Table Cell as Section [page 431]
Exporting a Report Element [page 432]
Deleting a Report Element [page 438]
Ordering a Report Element [page 439]
Working with Axes [page 440]
Working with Rankings [page 464]
Working with Calculations [page 468]
Working with Custom Properties [page 473]
Managing Data Filters in Report Elements [page 478]
Working with Sorts [page 484]
Managing Report Element Comments [page 493]

8.4.1 Creating a Report Element

Usage

Creates one of the following report elements:

- Cell (only free cells, not cells in a table)
- Section
- Table (VTable, HTable, XTable, and Form)
- Visualization (charts)
- Custom element
- From a shared element

> → Remember
>
> You cannot create PageZone elements and table elements of type Cell.

Request

`POST /documents/<documentID>/reports/<reportID>/elements?<optional_parameters>`

Optional Parameters

Optional Parameter	Description
`unit`	An optional parameter of type string that defines the unit of measurement used for all dimensional values such as size, padding, and position. Values are `metric` (default) , `inch` and `centimeter`.
`sharedelementId`	An integer parameter that specifies a valid identifier of a shared element in the CMS repository. Minimum is 1. The shared element is inserted into the document.

Request type: `application/xml` or `application/json`

Request body: the valid definition of an element. See the Chart Response Body Schema [page 70] in the case of a chart.

Some settings are common to all types of report elements, while some are only specific. These specific settings are located into `content` tags. The element is added and an ID attributed to the element.

There is no need of a request body when you create a report element from a shared element.

> → Remember
>
> - You can create a report element with its expressions or create an empty report element, and add expressions in a second call using `PUT /documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/expressions`.
> - You can change the type of report element after you create it by calling `PUT /documents/<documentID>/reports/<reportID>/elements/<elementID>`.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: Cell

`POST /documents/127/reports/1/elements?unit=inch`

In the request body, you define the cell dimensions, padding and contents.

```
<element type="Cell">
    <parentId>2</parentId>
    <size minimalWidth="2.23" minimalHeight="3.56" autofitWidth="false"
autofitHeight="true"/>
    <padding left="0.069" right="0.069" top="0.069" bottom="0.069"/>
    <content>
```

```
        <expression>
            <formula type="Text" dataType="Numeric">=Sum([Sales revenue])</
formula>
        </expression>
    </content>
</element>
```

Example: Section

```
POST /documents/127/reports/1/elements
```

In the request body, you define how duplicate row aggregation is managed and a formula.

```
<element type="Section">
    <parentId>2</parentId>
    <size minimalHeight="1000"/>
    <padding bottom="1000"/>
    <position repeatOnEveryVerticalPage="true"/>
    <style>
        <background>
            <color rgb="#0000ff"/>
        </background>
    </style>
    <content>
        <axes duplicateRowAggregation="true">
            <axis role="Row">
                <expressions>
                    <formula dataType="String">=[Year]</formula>
                </expressions>
            </axis>
        </axes>
    </content>
</element>
```

Example: Table

```
POST /documents/14695/reports/1/elements
```

Request body:

```
<element type="XTable">
    <parentId>2</parentId>
    <content>
        <axes>
            <axis role="Row">
                <expressions>
                    <formula dataType="String">=[Country]</formula>
                    <formula dataType="String">=[Year]</formula>
                </expressions>
            </axis>
            <axis role="Column">
                <expressions>
                    <formula dataType="String">=[Resort]</formula>
                    <formula dataType="String">=[Service]</formula>
                </expressions>
            </axis>
            <axis role="Body">
```

```
                  <expressions>
                     <formula dataType="Numeric">=[Number of guests]</formula>
                     <formula dataType="Numeric">=[Revenue]</formula>
                  </expressions>
            </axis>
        </axes>
    </content>
</element>
```

Example: Chart

```
POST /documents/13069/reports/6/elements?unit=centimeter
```

Request body:

```
<element type="Visualization">
    <parentId>2</parentId>
    <content>
        <chart type="HorizontalBar">
            <axes>
                <axis role="Color">
                    <expressions>
                        <formula dataType="String">=[Resort]</formula>
                        <formula dataType="String">=[Country]</formula>
                    </expressions>
                </axis>
                <axis role="Category">
                    <expressions>
                        <formula dataType="String">=[Year]</formula>
                    </expressions>
                </axis>
                <axis role="Value">
                    <expressions>
                        <formula dataType="Numeric">=[Revenue]</formula>
                    </expressions>
                </axis>
            </axes>
        </chart>
    </content>
</element>
```

Example: Empty Custom Element

Custom element type and server identifier are mandatory.

```
POST /documents/13060/reports/6/elements?unit=centimeter
```

Request body:

```
<element type="Custom">
    <parentId>2</parentId>
    <content>
        <custom type="google" serverId="Delegated_Rendering_Server">
        </custom>
    </content>
</element>
```

Example: Custom Element with Expressions

Custom element type and server identifier are mandatory.

```
POST /documents/13069/reports/6/elements
```

Request body:

```
<element type="Custom">
    <parentId>2</parentId>
    <content>
        <custom type="area" serverId="Delegated_Rendering_Server">
            <axes>
                <axis role="category-axis">
                    <expressions>
                        <formula dataType="String">=[Resort]</formula>
                        <formula dataType="String">=[Country]</formula>
                    </expressions>
                </axis>
                <axis role="region-color">
                    <expressions>
                        <formula dataType="String">=[Year]</formula>
                    </expressions>
                </axis>
                <axis role="primary-values">
                    <expressions>
                        <formula dataType="Numeric">=[Revenue]</formula>
                    </expressions>
                </axis>
            </axes>
        </custom>
    </content>
</element>
```

Example: From a Shared Element

```
POST /documents/8192/reports/2/elements?sharedelementId=8190
```

No request body.

Response:

```
<success>
    <message>The resource of type "Report element" with identifier "4" has been
successfully created.</message>
    <id>4</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the Details of a Report Element [page 400]

Managing Shared Elements [page 679]

8.4.2 Getting the List of Report Elements

Usage

Gets the elements of a specific report.

The following items identify a report element:

- Id
- Reference

Request

```
GET /documents/<documentID>/reports/<reportID>/elements?
unit=<unit>&allInfo=<boolean>
```

Where:

- `<unit>` is an optional parameter of type string that defines the unit of measurement used for all dimensional values such as size, padding, and position. Values are `metric` (default) , `inch` and `centimeter`.
- `<allInfo>` is an optional parameter of type=boolean (default=false). If set to true, the details of each report element is included. Available since 4.2 SP04.

Response

Response type: `application/xml` or `application/json`

Response body: the list of elements of the report, with the following information:

- `<id>`
- `<name>`
- `<reference>`
- `<size>`
- `<position>`
- `<padding>`
- `type` attribute, with the possible values `PageZone`, `Cell`, `VTable`, `HTable`, `XTable`, `Form`, `Visualization`, and `Custom`.

Example: XML

```
GET /documents/6528/reports/1/elements?unit=centimeter
```

Response:

```
<elements>
    <element type="Cell">
        <id>14</id>
        <reference>1.E</reference>
        <name>Block 2</name>
        <parentId>1</parentId>
        <size minimalWidth="7.772" minimalHeight="0.483" autofitWidth="false"
autofitHeight="true"/>
        <position x="15.322" y="0.104" horizontalAnchorType="None"
verticalAnchorType="None"
            oneHorizontalPage="false" newHorizontalPage="false"
repeatOnEveryVerticalPage="false"
            oneVerticalPage="false" newVerticalPage="false"/>
        <padding left="0.152" right="0.152" top="0.152" bottom="0.152"/>
    </element>
    <element type="Cell">
        <id>16</id>
        <reference>1.G</reference>
        <name>Block 2</name>
        <parentId>1</parentId>
        <size minimalWidth="0.132" minimalHeight="0.0" autofitWidth="true"
autofitHeight="true"/>
        <position x="1.35" y="1.111" horizontalAnchorType="None"
verticalAnchorType="None"
            oneHorizontalPage="false" newHorizontalPage="false"
repeatOnEveryVerticalPage="false"
            oneVerticalPage="false" newVerticalPage="false"/>
        <padding left="0.152" right="0.152" top="0.152" bottom="0.152"/>
    </element>
    <element type="PageZone">
        <id>1</id>
        <name>Header</name>
        <size minimalHeight="3.254"/>
    </element>
    <element type="XTable">
        <id>13</id>
        <reference>1.D</reference>
        <name>Block 1</name>
        <parentId>24</parentId>
        <position x="1.482" y="1.164" horizontalAnchorType="None"
verticalAnchorType="None"
            oneHorizontalPage="false" newHorizontalPage="false"
repeatOnEveryVerticalPage="false"
            oneVerticalPage="false" newVerticalPage="false"/>
        <padding left="0.0" right="0.0" top="0.0" bottom="0.0"/>
    </element>
    <element type="Cell">
        <id>9</id>
        <reference>1.9</reference>
        <parentId>13</parentId>
        <size minimalWidth="2.12" minimalHeight="0.4" autofitWidth="false"
autofitHeight="true"/>
        <padding left="0.176" right="0.176" top="0.155" bottom="0.162"/>
    </element>
    <element type="Section">
        <id>24</id>
        <reference>1.O</reference>
        <name>City</name>
        <parentId>2</parentId>
        <size minimalHeight="0.0"/>
```

```
        <position repeatOnEveryVerticalPage="false" oneVerticalPage="false"
newVerticalPage="false"/>
        <padding bottom="0.821"/>
    </element>
</elements>
```

Example: JSON

```
GET /documents/5157/reports/1/elements
```

Response:


```
{"elements":
    {"element":[
        {"@type":"PageZone",
         "id":1,
         "name":"Header",
         "size":{"@minimalHeight":"1984.0"}
        },
        {"@type":"Cell",
         "id":19,
         "reference":"1.J",
         "parentId":29,
         "size":{
             "@autofitHeight":"true",
             "@autofitWidth":"false",
             "@minimalHeight":"567.0",
             "@minimalWidth":"3005.0"},
         "padding":{
             "@bottom":"230.0",
             "@top":"220.0",
             "@right":"250.0",
             "@left":"250.0"}
        },
        ...
        {"@type":"HTable",
         "id":29,
         "reference":"1.T",
         "name":"Block 1 (1)",
         "parentId":2,
         "position":{
             "@newVerticalPage":"false",
             "@oneVerticalPage":"false",
             "@repeatOnEveryVerticalPage":"false",
             "@newHorizontalPage":"false",
             "@oneHorizontalPage":"false",
             "@verticalAnchorType":"None",
             "@horizontalAnchorType":"None",
             "@y":"1950.0",
             "@x":"2662.0"}
        },
        ...
        {"@type":"VTable",
         "id":16,
         "reference":"1.G",
         "name":"Block 1",
         "parentId":2,
         "position":{
             "@newVerticalPage":"false",
             "@oneVerticalPage":"false",
             "@repeatOnEveryVerticalPage":"false",
             "@newHorizontalPage":"false",
             "@oneHorizontalPage":"false",
```

```
            "@verticalAnchorType":"None",
            "@horizontalAnchorType":"None",
            "@y":"18450.0",
            "@x":"2700.0"}
        },
        ...
        {"@type":"XTable",
         "id":42,
         "reference":"1.g",
         "name":"Block 1 (2)",
         "parentId":2,
         "position":{
            "@newVerticalPage":"false",
            "@oneVerticalPage":"false",
            "@repeatOnEveryVerticalPage":"false",
            "@newHorizontalPage":"false",
            "@oneHorizontalPage":"false",
            "@verticalAnchorType":"None",
            "@horizontalAnchorType":"None",
            "@y":"7800.0",
            "@x":"2887.0"}
        }]
    }
}
```


Example: Using allInfo=true

```
GET /documents/6528/reports/1/elements?allInfo=true
```

Response:


```
<elements>
    <element type="PageZone">
        <id>2</id>
        <name>Header</name>
        <size minimalHeight="1984.0"/>
        <hide always="false"/>
        <style>
            <border>
                <top thickness="None" rgb="#000000" style="None"/>
                <bottom thickness="None" rgb="#000000" style="None"/>
                <left thickness="None" rgb="#000000" style="None"/>
                <right thickness="None" rgb="#000000" style="None"/>
            </border>
            <background>
                <color rgb="#ffffff"/>
            </background>
        </style>
    </element>
    <element type="VTable">
        <id>41</id>
        <reference>5.f</reference>
        <name>Block 1 (1)</name>
        <parentId>1</parentId>
        <position x="27337.0" y="262.0" horizontalAnchorType="None"
verticalAnchorType="None" oneHorizontalPage="false" newHorizontalPage="false"
repeatOnEveryVerticalPage="false" oneVerticalPage="false"
newVerticalPage="false"/>
        <hide always="false" whenEmpty="false"/>
        <style>
            <border>
                <top thickness="None" rgb="#000000" style="None"/>
                <bottom thickness="None" rgb="#000000" style="None"/>
```

```
                <left thickness="None" rgb="#000000" style="None"/>
                <right thickness="None" rgb="#000000" style="None"/>
            </border>
            <background width="0.0" height="0.0">
                <color rgb="#ffffff"/>
            </background>
            <alternateColor frequency="2" rgb="#f8fbfc"/>
        </style>
        <content>
            <axes duplicateRowAggregation="true">
                <axis role="Column">
                    <id>1</id>
                    <expressions>
                        ...
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.4.3 Getting the Details of a Report Element

Usage

Gets the details of a specific report element, referenced by its ID.

If the report element is a custom element and its service retuns a picture, the call returns a picture.

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>?
<optional_parameters>
```

Optional Parameters

Parameter	Description
`unit`	Specifies the unit of measurement used for all dimensional values such as size, padding, and position. Values are `metric` (default) , `inch` and `centimeter`.
`reference`	Specifies an element reference, for example: `{elementId}?reference=1.D.7`.
`datapath`	Since 4.2 SP04. Type=string. Specify a datapath to identify the instance of the report element. (ex: `/documents/7610/reports/1/elements/13?datapath=1.D.7)`

> **Note**
>
> You cannot specify `reference` and `datapath` in the same call. Only one may be specified.

Response:

Response type: `application/xml` or `application/json`

Response body: the definition of a report element, with the following information:

- `<id>`
- `<reference>`
- `<parentId>`
- `<size>`
- `<position>`
- `<hide>`
- `<padding>`
- `<style>`
- `<content>`
- `type` attribute, with the possible values `PageZone`, `Cell`, `VTable`, `HTable`, `XTable`, `Form`, `Visualization`, and `Custom`.
- `isLinkedToSharedElement` Boolean attribute, "true" if the report element is linked to a shared element, "false" otherwise.

Example: Cell

```
GET /documents/6580/reports/1/elements/4?unit=inch
```

Response:

```
<element type="Cell" isLinkedToSharedElement="false">
    <id>4</id>
    <reference>1.4</reference>
    <parentId>2</parentId>
    <size minimalWidth="4.18" minimalHeight="0.43" autofitWidth="false"
autofitHeight="true"/>
    <position x="2.06" y="0.12" horizontalAnchorType="None"
verticalAnchorType="None"
        oneHorizontalPage="false" newHorizontalPage="false"
repeatOnEveryVerticalPage="false"
        oneVerticalPage="false" newVerticalPage="false"/>
    <hide always="false"/>
    <padding left="0.059" right="0.059" top="0.043" bottom="0.043"/>
    <style>
        <border>
            <top thickness="None" rgb="#000000" style="None"/>
            <bottom thickness="None" rgb="#000000" style="None"/>
            <left thickness="None" rgb="#000000" style="None"/>
            <right thickness="None" rgb="#000000" style="None"/>
        </border>
```

```
        <background>
            <color rgb="#ff00ff"/>
        </background>
        <font size="16" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#333333"/>
        <alignment horizontal="Center" vertical="Center" wrapText="false"/>
    </style>
    <content>
        <expression>
            <formula type="Text" dataType="Numeric">=[帳票_累計(内)_発生]+[帳票_累計
(外)_発生]</formula>
            <format type="Custom">
                <template positive="#,##0" negative="-#,##0" zero="0"
undefined="0"/>
            </format>
        </expression>
        <alerters>
            <id>2</id>
        </alerters>
    </content>
</element>
```

Example: PageZone

```
GET /documents/6580/reports/1/elements/1?unit=centimeter
```

Response:

```
<element type="PageZone" isLinkedToSharedElement="false">
    <id>1</id>
    <size minimalHeight="1.4"/>
    <hide always="false"/>
    <style>
        <border>
            <top thickness="None" rgb="#000000" style="None"/>
            <bottom thickness="None" rgb="#000000" style="None"/>
            <left thickness="None" rgb="#000000" style="None"/>
            <right thickness="None" rgb="#000000" style="None"/>
        </border>
        <background>
            <color rgb="#ffffff"/>
        </background>
    </style>
</element>
```

Example: Section

```
GET /documents/6580/reports/1/elements/13?unit=inch
```

Response:

```
<element type="Section" isLinkedToSharedElement="false">
    <id>13</id>
    <reference>1.D</reference>
    <parentId>2</parentId>
    <size minimalHeight="0.2"/>
    <position repeatOnEveryVerticalPage="true" oneVerticalPage="true"
newVerticalPage="true"/>
```

```
    <hide always="false">
        <expression dataType="Boolean">=Min([Number of guests])=0</expression>
    </hide>
    <padding bottom="0.1"/>
    <style>
        <background>
            <color rgb="#ffffce"/>
        </background>
    </style>
    <content>
        <axes duplicateRowAggregation="true">
            <axis role="Row">
                <id>0</id>
                <expressions>
                    <formula dataType="String"
dataObjectId="DP0.DO39">=[Country]</formula>
                </expressions>
            </axis>
        </axes>
        <emptyHidingBlocks>
            <child id="14"/>
            <child id="11"/>
        </emptyHidingBlocks>
        <body>
            <child id="14"/>
            <child id="11"/>
        </body>
    </content>
</element>
```

Example: XTable

```
GET /documents/19035/reports/2/elements/12?unit=centimeter
```

Response:

```
<element type="XTable" isLinkedToSharedElement="false">
    <id>12</id>
    <reference>2.C</reference>
    <name>Block 1</name>
    <parentId>1</parentId>
    <position x="4.894" y="1.376" horizontalAnchorType="None"
verticalAnchorType="None"
        oneHorizontalPage="false" newHorizontalPage="false"
repeatOnEveryVerticalPage="false"
        oneVerticalPage="false" newVerticalPage="false"/>
    <hide always="false"/>
    <style>
        <border>
            <top thickness="None" rgb="#000000" style="None"/>
            <bottom thickness="None" rgb="#000000" style="None"/>
            <left thickness="None" rgb="#000000" style="None"/>
            <right thickness="None" rgb="#000000" style="None"/>
        </border>
        <background width="0.0" height="0.0"/>
        <alternateColor frequency="2" rgb="#fcfdfd"/>
    </style>
    <content>
        <axes duplicateRowAggregation="true">
            <axis role="Row">
                <id>0</id>
                <breaks>
```

```
                    <break onePage="false" newPage="false" sort="true"
duplicate="Remove"
                        repeatHeader="false" showHeader="true"
showFooter="true">
                        <id>1</id>
                        <formula dataType="String"
dataObjectId="DP0.DO31">=[Year]</formula>
                    </break>
                </breaks>
                <expressions>
                    <formula dataType="String" dataObjectId="DP0.DO31">=[Year]</
formula>
                    <formula dataType="String"
dataObjectId="DP0.DO39">=[Country]</formula>
                </expressions>
            </axis>
            <axis role="Column">
                <id>1</id>
                <expressions>
                    <formula dataType="String" dataObjectId="DP0.DO2">=[Resort]</
formula>
                </expressions>
            </axis>
            <axis role="Body">
                <id>2</id>
                <expressions>
                    <formula dataType="Numeric"
dataObjectId="DP0.DO7">=[Revenue]</formula>
                </expressions>
            </axis>
        </axes>
        <layout>
            <zone horizontalType="Body" verticalType="Body">
                <child row="0" rowSpan="1" column="0" columnSpan="1" id="5"/>
            </zone>
            <zone horizontalType="Body" verticalType="Header">
                <child row="0" rowSpan="1" column="0" columnSpan="1" id="8"/>
            </zone>
            <zone horizontalType="Body" verticalType="Header"
verticalBreakId="1">
                <child row="0" rowSpan="1" column="0" columnSpan="1" id="19"/>
            </zone>
            <zone horizontalType="Body" verticalType="Footer"
verticalBreakId="1">
                <child row="0" rowSpan="1" column="0" columnSpan="1" id="24"/>
                <child row="1" rowSpan="1" column="0" columnSpan="1" id="27"/>
            </zone>
            <zone horizontalType="Header" verticalType="Body">
                <child row="0" rowSpan="1" column="0" columnSpan="1" id="14"/>
                <child row="0" rowSpan="1" column="1" columnSpan="1" id="7"/>
            </zone>
            <zone horizontalType="Header" verticalType="Header">
                <child row="0" rowSpan="1" column="0" columnSpan="1" id="10"/>
                <child row="0" rowSpan="1" column="1" columnSpan="1" id="11"/>
            </zone>
            <zone horizontalType="Header" verticalType="Header"
verticalBreakId="1">
                <child row="0" rowSpan="1" column="0" columnSpan="1" id="16"/>
                <child row="0" rowSpan="1" column="1" columnSpan="1" id="17"/>
            </zone>
            <zone horizontalType="Header" verticalType="Footer"
verticalBreakId="1">
                <child row="0" rowSpan="1" column="0" columnSpan="1" id="21"/>
                <child row="0" rowSpan="1" column="1" columnSpan="1" id="22"/>
                <child row="1" rowSpan="1" column="0" columnSpan="1" id="25"/>
                <child row="1" rowSpan="1" column="1" columnSpan="1" id="26"/>
            </zone>
        </layout>
```

```
    </content>
</element>
```

Example: Cell with Reference

```
GET /documents/8362/reports/1/elements/6?reference=1.6.Ga
```

Response:

```
<element type="Cell" isLinkedToSharedElement="false">
    <id>6</id>
    <reference>1.6</reference>
    <parentId>11</parentId>
    <datapath>DP0.DO1:"Robert King"|"0000000007",DP0.DO10:" 6561 - Centre
Sport"|"0000006561"</datapath>
    ...
</element>
```

Example: Chart

```
GET /documents/6503/reports/1/elements/39?unit=inch
```

Response:

```
<element type="Visualization" isLinkedToSharedElement="false">
    <id>39</id>
    <reference>1.d</reference>
    <name>Block 2 (2)</name>
    <parentId>2</parentId>
    <size minimalWidth="4.165" minimalHeight="3.125"/>
    <position x="0.334" y="0.018" oneHorizontalPage="false"
newHorizontalPage="false"
        repeatOnEveryVerticalPage="false" oneVerticalPage="false"
newVerticalPage="false"/>
    <hide always="false"/>
    <style>
        <border>
            <top thickness="None" rgb="#000000" style="None"/>
            <bottom thickness="None" rgb="#000000" style="None"/>
            <left thickness="None" rgb="#000000" style="None"/>
            <right thickness="Thin" rgb="#ff0000" style="Dashed"/>
        </border>
        <background>
            <gradient orientation="Horizontal">
                <start alpha="255" rgb="#ffff00"/>
                <end alpha="255" rgb="#ff00ff"/>
            </gradient>
        </background>
    </style>
    <content>
        <chart type="HorizontalBar">
            <layout showDimensionsWithEmptyMeasureValues="true"
showDimensionsWithMeasuresEqualToZero="false"
                showDimensionsWithSumOfMeasuresEqualToZero="true"
showMeasuresWithEmptyDimensionValues="true"
                showParentNodes="true" duplicateRowAggregation="true"
horizontal="true"/>
            <title visible="true">
```

```
                <style>
                    <border thickness="None">
                        <color alpha="255" rgb="#000000"/>
                    </border>
                    <background>
                        <color alpha="0" rgb="#000000"/>
                    </background>
                    <font size="9" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#000000"/>
                    <alignment horizontal="Left" vertical="Center"
textPolicy="Truncate"/>
                </style>
                <label dataType="String">="Bar"</label>
                <layout location="Top" orientation="Auto" spacing="2"
adjust="false"/>
            </title>
            <legend visible="true">
                <style>
                    <border thickness="None">
                        <color alpha="0" rgb="#000000"/>
                    </border>
                    <background>
                        <color alpha="0" rgb="#000000"/>
                    </background>
                    <font size="8" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#707070"/>
                    <alignment horizontal="Left" vertical="Center"
textPolicy="Wrap"/>
                </style>
                <title visible="true">
                    <style>
                        <border thickness="None">
                            <color alpha="255" rgb="#000000"/>
                        </border>
                        <background>
                            <color alpha="0" rgb="#000000"/>
                        </background>
                        <font size="8" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#000000"/>
                        <alignment horizontal="Left" vertical="Center"
textPolicy="Truncate"/>
                    </style>
                    <layout orientation="Auto" spacing="2"/>
                </title>
                <layout groupByDimension="false" symbolSize="9" location="Right"
orientation="Auto" spacing="4" adjust="false"/>
            </legend>
            <dataLabels type="Auto" visible="false">
                <style>
                    <border thickness="None">
                        <color alpha="255" rgb="#000000"/>
                    </border>
                    <background>
                        <color alpha="0" rgb="#000000"/>
                    </background>
                    <font size="8" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#707070"/>
                </style>
                <layout autoHiding="true" resolveOverlapping="false"
position="InsideFirstOutsideOtherwise" orientation="Vertical" spacing="0"/>
            </dataLabels>
            <plotArea>
                <barSeries dashedLines="false" spacingBetweenGroups="0.2"
spacingWithinGroups="0.2"/>
                <background mode="Plain">
                    <coloring>
                        <color alpha="0" rgb="#000000"/>
                    </coloring>
```

```
                </background>
                <grids>
                    <grid type="Vertical">
                        <color alpha="0" rgb="#000000"/>
                    </grid>
                    <grid type="Horizontal">
                        <color alpha="255" rgb="#e7e7e7"/>
                    </grid>
                </grids>
            </plotArea>
            <graphics>
                <coloring>
                    <palettes>
                        <palette alpha="0" refId="green"/>
                    </palettes>
                </coloring>
                <rendering filter="RealLightingAndComplexShadows"
look3D="false">
                    <edge type="None"/>
                    <shadow effect="OneSided" xOffset="0.0" yOffset="0.0"
filterPassCount="3" filterWindowSize="5" lightPower="0.3">
                        <color alpha="187" rgb="#9d9d9d"/>
                    </shadow>
                </rendering>
                <effects>
                    <barSeries effect="None" roundedCorners="false"/>
                </effects>
            </graphics>
            <axes>
                <axis role="Color" optional="true">
                    <id>0</id>
                    <name>Region Color</name>
                    <expressions>
                        <formula dataType="String"
dataObjectId="DP0.DO4">=[Service Line]</formula>
                    </expressions>
                </axis>
                <axis role="Category" visible="true" optional="false">
                    <id>1</id>
                    <name>Category Axis</name>
                    <title visible="true">
                        <style>
                            <border thickness="None">
                                <color alpha="255" rgb="#000000"/>
                            </border>
                            <background>
                                <color alpha="0" rgb="#000000"/>
                            </background>
                            <font size="8" face="Arial" italic="false"
bold="true" strikethrough="false" underline="false" rgb="#000000"/>
                        </style>
                        <layout spacing="0"/>
                        <separator>&amp;amp;</separator>
                    </title>
                    <layout reverseOrder="false" continuous="false"
adjust="false"/>
                    <coloring>
                        <color alpha="255" rgb="#707070"/>
                    </coloring>
                    <grid>
                        <color alpha="255" rgb="#dadada"/>
                        <background>
                            <color alpha="0" rgb="#000000"/>
                        </background>
                    </grid>
                    <tick length="4" margin="0"/>
                    <labels visible="true" orientation="Auto" deleteMode="Auto"
autoResize="false" staggered="false">
```

```
                        <font size="8" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#707070"/>
                    </labels>
                    <expressions>
                        <formula dataType="String"
dataObjectId="DP0.DO39">=[Country]</formula>
                    </expressions>
                </axis>
                <axis role="Value" visible="true" optional="false">
                    <id>2</id>
                    <name>Value Axis 1</name>
                    <title visible="true">
                        <style>
                            <border thickness="None">
                                <color alpha="255" rgb="#000000"/>
                            </border>
                            <background>
                                <color alpha="0" rgb="#000000"/>
                            </background>
                            <font size="8" face="Arial" italic="false"
bold="true" strikethrough="false" underline="false" rgb="#000000"/>
                        </style>
                        <layout spacing="0"/>
                        <separator>&amp;amp;</separator>
                    </title>
                    <layout adjust="false"/>
                    <coloring>
                        <color alpha="255" rgb="#707070"/>
                    </coloring>
                    <tick length="5" margin="4" densityMode="Fixed"
density="2"/>
                    <labels visible="true" orientation="Auto" staggered="false">
                        <font size="8" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#707070"/>
                    </labels>
                    <stacking mode="GloballyStacked" stacked100percent="true"/>
                    <scaling mode="Linear" originInRange="Always"
unitScaleFactor="0" roundMinMaxValues="false">
                        <minValue type="Auto"/>
                        <maxValue type="Auto"/>
                    </scaling>
                    <expressions>
                        <formula regionType="Default" dataType="Numeric"
dataObjectId="DP0.DO7">=[Revenue]</formula>
                    </expressions>
                </axis>
            </axes>
        </chart>
    </content>
</element>
```


Example: Custom Element

```
GET /documents/6580/reports/1/elements/138?unit=inch
```

Response


```
<element type="Custom">
    <id>138</id>
    <reference>2.4</reference>
    <parentId>2</parentId>
    <size minimalWidth="9.141" minimalHeight="2.473"/>
```

```
    <position x="0.25" y="0.55" horizontalAnchorType="None"
verticalAnchorType="None" oneHorizontalPage="false" newHorizontalPage="false"
repeatOnEveryVerticalPage="false" oneVerticalPage="false"
newVerticalPage="false"/>
    <hide always="false" whenEmpty="false"/>
    <style>
        <border>
            <top thickness="None" rgb="#000000" style="None"/>
            <bottom thickness="None" rgb="#000000" style="Dotted"/>
            <left thickness="None" rgb="#00ee00" style="None"/>
            <right thickness="None" rgb="#000000" style="None"/>
        </border>
    </style>
    <content>
        <custom type="area" serverId="Delegated_Rendering_Server">
            <layout showDimensionsWithEmptyMeasureValues="true"
showDimensionsWithMeasuresEqualToZero="true"
showDimensionsWithSumOfMeasuresEqualToZero="true"
showMeasuresWithEmptyDimensionValues="true" duplicateRowAggregation="false"/>
            <name>CVOM HTML5 area chart</name>
            <description>Description for Area</description>
            <axes>
                <axis role="category-axis" optional="false">
                    <id>0</id>
                    <name>Category Axis</name>
                    <expressions>
                        <formula dataType="String"
dataObjectId="DP0.DO2">=[Resort]</formula>
                        <formula dataType="String"
dataObjectId="DP0.DOa5">=[Region]</formula>
                        <formula dataType="String"
dataObjectId="DP0.DOa7">=[City]</formula>
                    </expressions>
                </axis>
                <axis role="region-color" optional="true">
                    <id>1</id>
                    <name>Color</name>
                    <expressions>
                        <formula dataType="String"
dataObjectId="DP0.DO18">=[Year]</formula>
                    </expressions>
                </axis>
                <axis role="primary-values" optional="true">
                    <id>2</id>
                    <name>Values</name>
                    <expressions>
                        <formula dataType="Numeric"
dataObjectId="DP0.DO25">=[Future guests]</formula>
                    </expressions>
                </axis>
            </axes>
        </custom>
    </content>
</element>
```

Example: Custom Element using the "fold" attribute

Since 4.2 SP04, a 'fold' attribute specifies if the axis/axis's break of the block or the instance of the block is folded.

Response

```
<element type="HTable" isLinkedToSharedElement="false">
```

```
    <id>75</id>
    <reference>m.1B.1.3</reference>
    <name>Block 3</name>
    <parentId>65</parentId>
    <datapath>DP0.DObc:"2005",DP0.DOda:"Florida"</datapath>
    <position x="20737.0" y="787.0" horizontalAnchorType="None"
verticalAnchorType="None" oneHorizontalPage="false" newHorizontalPage="false"
repeatOnEveryVerticalPage="false" oneVerticalPage="false"
newVerticalPage="false"/>
    <hide always="false" whenEmpty="false"/>
    <style>
        <border>
            <top thickness="None" rgb="#000000" style="None"/>
            <bottom thickness="None" rgb="#000000" style="None"/>
            <left thickness="None" rgb="#000000" style="None"/>
            <right thickness="None" rgb="#000000" style="None"/>
        </border>
        <background width="0.0" height="0.0">
            <color rgb="#ffffff"/>
        </background>
        <alternateColor frequency="2" rgb="#f8fbfc"/>
    </style>
    <content>
        <axes duplicateRowAggregation="true">
            <axis role="Row" fold="true">
                <id>0</id>
                <expressions>
                    <formula dataType="String" dataObjectId="DP0.DOa5">=[Lines]</
formula>
                    <formula dataType="Numeric" dataObjectId="DP0.DO93">=[Sales
revenue]</formula>
                </expressions>
            </axis>
        </axes>
        <layout showColumnsWithEmptyMeasureValues="true"
showColumnsWithMeasuresEqualToZero="true"
showColumnsWithSumOfMeasuresEqualToZero="true"
showRowsColumnsWithEmptyDimensionValues="false" showHeader="true"
showFooter="false" repeatHeaderHorizontally="false">
            <zone horizontalType="Body" verticalType="Body">
                <child row="0" rowSpan="1" column="0" columnSpan="1" id="76"/>
                <child row="1" rowSpan="1" column="0" columnSpan="1" id="77"/>
            </zone>
            <zone horizontalType="Header" verticalType="Body">
                <child row="0" rowSpan="1" column="0" columnSpan="1" id="79"/>
                <child row="1" rowSpan="1" column="0" columnSpan="1" id="80"/>
            </zone>
        </layout>
    </content>
</element>
```


Example: Specifying the optional data path

```
GET /documents/7409/reports/1/elements/24?
datapath=DP0.DO84%3A%22US%22%2CDP0.DOa7%3A%22Washington+D.C.%22
```

Response:


```
<element type="Cell" isLinkedToSharedElement="false">
    <id>24</id>
    <reference>1.O.6</reference>
    <parentId>23</parentId>
    <datapath>DP0.DO84:"US",DP0.DOa7:"Washington D.C."</datapath>
```

```
    . . .
</element>
```

Related Information

Charts [page 69]
Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]
Exporting a Report Element [page 432]

8.4.4 Getting the Background of a Report Element

Usage

Returns the background image or skin of a report element as a binary stream.

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/background
```

Response

The response type depends on the background image or skin format, for example:

- `image/bmp`
- `image/gif`
- `image/png`
- `image/jpg`
- `image/jpeg`

Use `image/*` if you do not know the background format.

An error is thrown when:

- There is no background.
- Response type and background format are not compatible.

Example

```
GET /documents/9199/reports/154769/elements/289477/background
```

Response:

```
Content-Type: image/png
HTTP Response Code: 200
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.5 Getting the Datapaths of a Report Element

Usage

Gets the datapaths available for a report element.

A datapath allows you to find a specific instance of a report element that can be used for example in several sections, or to find a specific report element of type cell in a table. The datapath may also be used to export a report element or to get its associated data.

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/datapaths?
limit=10&offset=0<optional_parameters>
```

Where:

- `limit` is an integer that specifies the number of datapaths to return. Values range from 0 to 50. If not provided, it takes a default value of 10.
- `offset` is an integer that offsets the beginning of the list. There is no maximum value. If not provided, it takes a default value of 0.

Optional Parameters

Parameter	Description
`datapath`	A string that specifies an initial datapath needed to get to the next level, for example `datapaths?datapath=DP0.DO18:"FY93"`. **Note** This parameter must be URL encoded.
`reference`	A string that specifies a reference, for example `datapaths?reference=1.D.7`.

Caution

You cannot use `datapath` and `reference` parameters in the same URL. This generates a 400 HTTP error.

Response

Response type: `application/xml` or `application/json`

Example: No Datapath

```
GET /documents/8413/reports/1/elements/20/datapaths?limit=20&offset=0
```

Response:

```
<datapaths elementId="17">
    <datapath>DP0.DObc:"2004"</datapath>
    <datapath>DP0.DObc:"2005"</datapath>
    <datapath>DP0.DObc:"2006"</datapath>
</datapaths>
```

Note

The returned `elementId` attribute specifies the report element identifier on which to go on iterating.

```
GET /documents/7070/reports/217/elements/227/datapaths?limit=10&offset=0
```

```
<datapaths elementId="227">

<datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q1",DP0.DOa5:"Accessories"<
/datapath>
    <datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q1",DP0.DOa5:"City
Skirts"</datapath>
    <datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q1",DP0.DOa5:"City
Trousers"</datapath>

<datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q1",DP0.DOa5:"Dresses"</
datapath>

<datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q1",DP0.DOa5:"Jackets"</
datapath>
```

```
<datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q1",DP0.DOa5:"Leather"</
datapath>

<datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q1",DP0.DOa5:"Outerwear"</
datapath>

<datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q1",DP0.DOa5:"Overcoats"</
datapath>
    <datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q1",DP0.DOa5:"Shirt
Waist"</datapath>

<datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q1",DP0.DOa5:"Sweaters"</
datapath>
</datapaths>
```

```
GET /documents/7070/reports/217/elements/227/datapaths?limit=10&offset=10
```

```
<datapaths elementId="227">
    <datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q1",DP0.DOa5:"Sweat-T-
Shirts"</datapath>

<datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q1",DP0.DOa5:"Trousers"</
datapath>

<datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q2",DP0.DOa5:"Accessories"<
/datapath>
    <datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q2",DP0.DOa5:"City
Skirts"</datapath>
    <datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q2",DP0.DOa5:"City
Trousers"</datapath>

<datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q2",DP0.DOa5:"Dresses"</
datapath>

<datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q2",DP0.DOa5:"Jackets"</
datapath>

<datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q2",DP0.DOa5:"Outerwear"</
datapath>

<datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q2",DP0.DOa5:"Overcoats"</
datapath>
    <datapath>DP0.DObc:"2014",DP0.DOa6:"Austin",DP0.DOba:"Q2",DP0.DOa5:"Shirt
Waist"</datapath>
</datapaths>
```

Example: With a Datapath

```
GET /documents/8413/reports/1/elements/17/datapaths?
datapath=DP0.DOa5%3A%22Sweaters%22%2CDP0.DOda%3A%22Texas%22%2CDP0.DObc%3A%222006%22
&limit=30&offset=0
```

Response:

```
<datapaths elementId="10">
    <datapath>DP0.DObc:"2006",DP0.DOda:"California"</datapath>
    <datapath>DP0.DObc:"2006",DP0.DOda:"Colorado"</datapath>
    <datapath>DP0.DObc:"2006",DP0.DOda:"DC"</datapath>
    <datapath>DP0.DObc:"2006",DP0.DOda:"Florida"</datapath>
    <datapath>DP0.DObc:"2006",DP0.DOda:"Illinois"</datapath>
```

```
    <datapath>DP0.DObc:"2006",DP0.DOda:"Massachusetts"</datapath>
    <datapath>DP0.DObc:"2006",DP0.DOda:"New York"</datapath>
    <datapath>DP0.DObc:"2006",DP0.DOda:"Texas"</datapath>
 </datapaths>
```

Example: With a Reference

```
GET /documents/8413/reports/1/elements/10/datapaths?reference=1.5.2.7
```

Response:

```
<datapaths elementId="11">
    <datapath>DP0.DOa5:"Accessories"</datapath>
    <datapath>DP0.DOa5:"City Skirts"</datapath>
    <datapath>DP0.DOa5:"City Trousers"</datapath>
    <datapath>DP0.DOa5:"Dresses"</datapath>
    <datapath>DP0.DOa5:"Jackets"</datapath>
    <datapath>DP0.DOa5:"Leather"</datapath>
    <datapath>DP0.DOa5:"Outerwear"</datapath>
    <datapath>DP0.DOa5:"Overcoats"</datapath>
    <datapath>DP0.DOa5:"Shirt Waist"</datapath>
    <datapath>DP0.DOa5:"Sweaters"</datapath>
    <datapath>DP0.DOa5:"Sweat-T-Shirts"</datapath>
    <datapath>DP0.DOa5:"Trousers"</datapath>
 </datapaths>
```

Example: Workflow

1. `GET /documents/8275/reports/1/elements/16/dataset` to get chart data (ID=16).
   This returns an error:

   ```
   <error>
       <error_code>101</error_code>
       <message>The resource of type "Report element" with identifier "16"
   contains datapaths: one must be
           specified.</message>
   </error>
   ```

2. `GET /documents/8275/reports/1/elements/16/datapaths` to get a datapath to access values.
   This returns:

   ```
   <datapaths elementId="5">
       <datapath>DP0.DO18:"FY92"</datapath>
       <datapath>DP0.DO18:"FY93"</datapath>
       <datapath>DP0.DO18:"FY94"</datapath>
       <datapath>DP0.DO18:"FY95"</datapath>
   </datapaths>
   ```

3. `GET /documents/8275/reports/1/elements/5/datapaths?datapath=DP0.DO18:"FY92"` to get the needed datapath, given the elementId and the chosen datapath:

   ```
   <datapaths elementId="8">
       <datapath>DP0.DO18:"FY92",DP0.DO1a:"Q1"</datapath>
       <datapath>DP0.DO18:"FY92",DP0.DO1a:"Q2"</datapath>
       <datapath>DP0.DO18:"FY92",DP0.DO1a:"Q3"</datapath>
   ```

```
    <datapath>DP0.DO18:"FY92",DP0.DO1a:"Q4"</datapath>
</datapaths>
```

4. `GET /documents/8275/reports/1/elements/8/datapaths?datapath=DP0.DO18:"FY92",DP0.DO1a:"Q1"` to iterate until the datapath list is empty:

```
<datapaths/>
```

5. `GET /documents/8275/reports/1/elements/16/dataset?datapath=DP0.DO18:"FY92",DP0.DO1a:"Q1"` to get the chart data using the chosen datapath:

```
<dataset>
    <data id="Category" type="String">
        <label>Month</label>
        <value>Feb</value>
    </data>
    <data id="Value" type="Numeric">
        <label>Revenue</label>
        <value>291190</value>
    </data>
</dataset>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]
Getting the Dataset of a Report Element [page 416]

8.4.6 Getting the Dataset of a Report Element

Usage

Returns the data of a report element of type Cell, Table, Visualization (charts), or custom element.

You may use a specific datapath or reference of a report element to retrieve its data.

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/dataset?
<optional_parameters>
```

Optional Parameters

Parameter	Description
`datapath`	A string that specifies a datapath, which allows you to find a specific instance of a report element, for example `dataset?datapath=DP1.DObc:"2004",DP1.DOa6:"Austin"`.
`reference`	A string that specifies a reference, for example `dataset?reference=1.D.7`.

> ⚠ Caution
>
> You cannot use `datapath` and `reference` parameters in the same URL. This generates a 400 HTTP error.

Response

Response type: `application/xml` or `application/json`

Response body: the details of the dataset, which are:

- The column metadata (column name, type, and data object identifier)
- A list of rows containing the values

Example: Cell

```
GET /documents/6404/reports/4/elements/7/dataset?reference=4.7.B
```

Response:

```
<dataset>
    <metadata>
        <value dataObjectId="DP1.DO93" type="Numeric">Sales revenue</value>
    </metadata>
    <row>
        <value>910451.2</value>
    </row>
</dataset>
```

Example: VTable

The vertical table contains six rows which can display up to five data object values.

```
GET /documents/6395/reports/1/elements/78/dataset
```

Response:

```
<dataset>
    <metadata>
        <value dataObjectId="DP1.DO84" type="String">Country</value>
        <value dataObjectId="DP1.DO2" type="String">Resort</value>
```

```
        <value dataObjectId="DP1.DO5" type="String">Service</value>
        <value dataObjectId="DP1.DO14" type="Numeric">Number of guests</value>
        <value dataObjectId="DP1.DO7" type="Numeric">Revenue</value>
    </metadata>
    <row>
        <value>Bahamas Islands</value>
        <value>Bahamas Beach</value>
        <value>Activities</value>
        <value/>
        <value>65600</value>
    </row>
    <row>
        <value>Bahamas Islands</value>
        <value>Bahamas Beach</value>
        <value>Bungalow</value>
        <value>48</value>
        <value>47040</value>
    </row>
    <row>
        <value>Bahamas Islands</value>
        <value>Sun Resort</value>
        <value>Activities</value>
        <value/>
        <value>9000</value>
     </row>
     <row>
        <value>Bahamas Islands</value>
        <value>Sun Resort</value>
        <value>Bungalow</value>
        <value>44</value>
        <value>40800</value>
     </row>
     <row>
        <value>Bahamas Islands</value>
        <value>Hawaiian Club</value>
        <value>Activities</value>
        <value/>
        <value>101100</value>
     </row>
     <row>
        <value>Bahamas Islands</value>
        <value>Hawaiian Club</value>
        <value>Bungalow</value>
        <value>45</value>
        <value>56100</value>
     </row>
</dataset>
```


Example: HTable

The horizontal table contains six columns that can display up to three data object values.

```
GET /documents/6361/reports/3/elements/101/dataset
```

Response (JSON):


```
{"dataset":
    {"metadata":
        {"value":
            [{"@dataObjectId":"DP0.DOa5",
              "@type":"String",
              "$":"Region"},
             {"@dataObjectId":"DP0.DOa7",
```

```
                "@type":"String",
                "$":"City"},
               {"@dataObjectId":"DP0.DO14",
                "@type":"Numeric",
                "$":"Number of guests"}]
            },
        "row":
            [{"value":["Bavaria","Augsburg",191]},
             {"value":["Bavaria","Munich",132]},
             {"value":["East Germany","Berlin",6]},
             {"value":["East Germany","Dresden",4]},
             {"value":["East Germany","Magdeburg",133]},
             {"value":["Ruhr","Cologne",4]}]
    }
}
```


Example: XTable

The cross table displays the values of six data objects.

```
GET /documents/6395/reports/3/elements/19/dataset
```

Response:


```
<dataset>
    <metadata>
        <value dataObjectId="DP1.DO84" type="String">Country</value>
        <value dataObjectId="DP1.DOa7" type="String">City</value>
        <value dataObjectId="DP1.DO2" type="String">Resort</value>
        <value dataObjectId="DP1.DO5" type="String">Service</value>
        <value dataObjectId="DP1.DO7" type="Numeric">Revenue</value>
        <value dataObjectId="DP1.DO14" type="Numeric">Number of guests</value>
    </metadata>
    <row>
        <value>Bahamas Islands</value>
        <value>Coral Bay</value>
        <value>Bahamas Beach</value>
        <value>Activities</value>
        <value>5600</value>
        <value/>
    </row>
    <row>
        <value>Bahamas Islands</value>
        <value>Long Island</value>
        <value>Bahamas Beach</value>
        <value>Bungalow</value>
        <value>3200</value>
        <value>4</value>
    </row>
    <row>
        <value>Bahamas Islands</value>
        <value>Nassau</value>
        <value>Bahamas Beach</value>
        <value>Bungalow</value>
        <value>43840</value>
        <value>44</value>
    </row>
    <row>
        <value>Bahamas Islands</value>
        <value>Nassau</value>
        <value>Sun Resort</value>
        <value>Bungalow</value>
        <value>40800</value>
```

```
        <value>44</value>
    </row>
    <row>
        <value>Bahamas Islands</value>
        <value>Nassau</value>
        <value>Hawaiian Club</value>
        <value>Bungalow</value>
        <value>56100</value>
        <value>45</value>
    </row>
    <row>
        <value>Bahamas Islands</value>
        <value>San Salvador</value>
        <value>Bahamas Beach</value>
        <value>Activities</value>
        <value>60000</value>
        <value/>
    </row>
    <row>
        <value>Bahamas Islands</value>
        <value>San Salvador</value>
        <value>Sun Resort</value>
        <value>Activities</value>
        <value>9000</value>
        <value/>
    </row>
    <row>
        <value>Bahamas Islands</value>
        <value>San Salvador</value>
        <value>Hawaiian Club</value>
        <value>Activities</value>
        <value>101100</value>
        <value/>
    </row>
</dataset>
```


Example: Chart

The chart displays three data object values.

```
GET /documents/6405/reports/1/elements/13/dataset?datapath=DP1.DOa6:"New York"
```

Response (XML):


```
<dataset>
    <metadata>
        <value dataObjectId="DP1.DO178" type="String">Store name</value>
        <value dataObjectId="DP1.DObc" type="String">Year</value>
        <value dataObjectId="DP1.DO93" type="Numeric">Sales revenue</value>
    </metadata>
    <row>
        <value>e-Fashion New York 5th</value>
        <value>2004</value>
        <value>644635.1</value>
    </row>
    <row>
        <value>e-Fashion New York 5th</value>
        <value>2005</value>
        <value>1.076144e6</value>
    </row>
    <row>
        <value>e-Fashion New York 5th</value>
        <value>2006</value>
```

```
        <value>1.2395874e6</value>
    </row>
    <row>
        <value>e-Fashion New York Magnolia</value>
        <value>2004</value>
        <value>1.0230607e6</value>
    </row>
    <row>
        <value>e-Fashion New York Magnolia</value>
        <value>2005</value>
        <value>1.6873591e6</value>
    </row>
    <row>
        <value>e-Fashion New York Magnolia</value>
        <value>2006</value>
        <value>1.9114343e6</value>
    </row>
</dataset>
```


Example: Chart

The chart displays five data object values.

`GET /documents/6404/reports/4/elements/16/dataset`

Response (JSON):


```
{"dataset":
    {"metadata":
        {"value":
            [{"@dataObjectId":"DP1.DOda","@type":"String","$":"State"},
             {"@dataObjectId":"DP1.DOa6","@type":"String","$":"City"},
             {"@dataObjectId":"DP1.DO178","@type":"String","$":"Store name"},
             {"@dataObjectId":"DP1.DO93","@type":"Numeric","$":"Sales revenue"},
             {"@dataObjectId":"DP1.DO94","@type":"Numeric","$":"Quantity sold"}]
        },
     "row":
        [{"value":["California","Los Angeles","e-Fashion Los
Angeles","1.6566757e6",9869]},
         {"value":["California","San Francisco","e-Fashion San
Francisco","1.3360033e6",7900]},
         {"value":["Colorado","Colorado Springs","e-Fashion Colorado
Springs",843584.2,5116]},
         {"value":["DC","Washington","e-Fashion Washington
Tolbooth","1.0535814e6",6491]},
         {"value":["Florida","Miami","e-Fashion Miami Sundance",811923.6,4830]},
         {"value":["Illinois","Chicago","e-Fashion Chicago
33rd","1.1340854e6",6519]},
         {"value":["Massachusetts","Boston","e-Fashion Boston
Newbury",887169.2,5269]},
         {"value":["New York","New York","e-Fashion New York
5th","1.2395874e6",7458]},
         {"value":["New York","New York","e-Fashion New York
Magnolia","1.9114343e6",11651]},
         {"value":["Texas ","Austin","e-Fashion Austin","1.1354791e6",6919]},
         {"value":["Texas ","Dallas","e-Fashion Dallas",803420.8,4932]},
         {"value":["Texas ","Houston","e-Fashion Houston",910451.2,5419]},
         {"value":["Texas ","Houston","e-Fashion Houston
Leighton","1.3357472e6",7923]}]
    }
}
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.7 Getting the Details of a Hierarchy Member

Usage

Gets the detail of a member in a hierarchical dimension.

Request

```
GET /documents/<documentId>/reports/<reportId>/elements/<elementId>/members/
<memberId>?<dataObjectID=string>
```

Where:

- `dataObjectID` is a string parameter that defines the data object if it is a chart.

Response

Response type: `application/xml` or `application/json`

Example (XML)

Example with a table cell

```
GET /documents/6797/reports/10/elements/15/members/0.0.1
```

Response body:

```
<member>
    <id>0.0.1</id>
    <level>8</level>
</member>
```

Example with a chart

```
GET /documents/6797/reports/10/elements/36/members/0.0.0?dataobjectId=DP0.DO87
```

Response body:

```
<member>
    <id>0.0.0</id>
    <level>5</level>
</member>
```

8.4.8 Updating a Report Element

Usage

- You can update the following report elements:
  - Cell
  - Section
  - Table (VTable, HTable, XTable, and Form)
  - PageZone
  - Visualization (charts)
  - Custom element
- You can change the type of the following report elements:
  - From a type of Table to another type of Table
  - From a Table to a Visualization
  - From a Visualization to a Table
  - From a type of Visualization to another type of Visualization
  - From a Table to a Custom Element
  - From a Visualization to a Custome Element
  - From a Custom Element to another Customer Element

Request

```
PUT /documents/<documentID>/reports/<reportID>/elements/<elementID>?unit=<unit>
```

Where:

- `<unit>` is an optional parameter of type string that defines the unit of measurement used for all dimensional values such as size, padding, and position. Values are `metric` (default) , `inch` and `centimeter`.

Request body: the relevant parameters for the element you want to update. See Charts [page 69] for a description of the Visualizations.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Updating the Axis Expressions

You can also update the axis expressions of a report element by using the following API:

`PUT /documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/expressions`

See Updating the Expressions of an Axis [page 446].

Example: Section

`PUT /documents/127/reports/1/elements/43`

Request body:

```
<element type="Section">
    <parentId>2</parentId>
    <content>
        <axes duplicateRowAggregation="true">
            <axis role="Row">
                <expressions>
                    <formula dataType="String">=[Year]</formula>
                </expressions>
            </axis>
        </axes>
    </content>
</element>
```

Response:

```
<success>
    <message>The resource of type "Report element" with identifier "43" has been
successfully updated.</message>
    <id>43</id>
</success>
```

Example: Cell

Request body:

```
<element type="Cell">
    <parentId>2</parentId>
    <size minimalWidth="4500" minimalHeight="675" autofitWidth="false"
autofitHeight="true"/>
```

```
    <padding left="75" right="75" top="75" bottom="75"/>
    <content>
        <expression>
            <formula type="Text" dataType="Numeric">=Sum([Sales revenue])</
formula>
        </expression>
    </content>
</element>
```

Example: VTable

Request body:

```
<element type="VTable">
    <parentId>2</parentId>
    <position x="8888" y="4444" horizontalAnchorType="None"
verticalAnchorType="None"/>
    <style>
        <border>
            <top thickness="None" rgb="#000000" style="None"/>
            <bottom thickness="None" rgb="#000000" style="None"/>
            <left thickness="None" rgb="#000000" style="None"/>
            <right thickness="None" rgb="#000000" style="None"/>
        </border>
        <background width="10" height="10"/>
        <alternateColor frequency="2" rgb="#fcfdfd"/>
    </style>
</element>
```

Example: PageZone

Request body:

> **Note**
>
> Only the following tags are available for PageZone report elements:
>
> - `<id>`
> - `<size>` with the `minimalHeight` attribute
> - `<hide>` without the associated expression
> - `<style>` (`<font>` and `<alignment>` tags are not available in a PageZone style.)

```
<element type="PageZone">
    <id>1</id>
    <size minimalHeight="8888"/>
    <hide always="true"/>
    <style>
        <border>
            <top thickness="Thin" rgb="#ff0000" style="Dotted"/>
            <bottom thickness="Thin" rgb="#ff0000" style="Dotted"/>
            <left thickness="Thin" rgb="#ff0000" style="Dotted"/>
            <right thickness="Thin" rgb="#ff0000" style="Dotted"/>
        </border>
        <background>
```

```
            <skin>Curve</skin>
        </background>
    </style>
</element>
```

Example: Chart

Request body:

```
<element type="Visualization">
    <parentId>2</parentId>
    <size minimalWidth="10.84" minimalHeight="6.26"/>
    <position x="0.2" y="0.2"/>
    <style>
        <border>
            <top thickness="None" rgb="#000000" style="None"/>
            <bottom thickness="None" rgb="#000000" style="None"/>
            <left thickness="None" rgb="#000000" style="None"/>
            <right thickness="None" rgb="#000000" style="None"/>
        </border>
        <background>
            <gradient orientation="Horizontal">
                <start alpha="255" rgb="#ffff00"/>
                <end alpha="255" rgb="#ff00ff"/>
            </gradient>
        </background>
    </style>
    <content>
        <chart type="TagCloud">
            <layout showDimensionsWithEmptyMeasureValues="true"
showDimensionsWithMeasuresEqualToZero="true"
                showDimensionsWithSumOfMeasuresEqualToZero="true"
showMeasuresWithEmptyDimensionValues="false"
                showParentNodes="true" duplicateRowAggregation="true"/>
            <title visible="true">
                <style>
                    <border thickness="None">
                        <color rgb="#000000" alpha="255"/>
                    </border>
                    <background>
                        <color rgb="#000000" alpha="0"/>
                    </background>
                    <font size="9" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#000000"/>
                    <alignment horizontal="Left" vertical="Center"
textPolicy="Truncate"/>
                </style>
                <label dataType="String">="Tag Cloud"</label>
                <layout location="Top" orientation="Auto" spacing="2"
adjust="false"/>
            </title>
            <legend visible="true">
                <style>
                    <border thickness="None">
                        <color rgb="#000000" alpha="0"/>
                    </border>
                    <background>
                        <color rgb="#000000" alpha="0"/>
                    </background>
                    <font size="8" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false" rgb="#707070"/>
                    <alignment horizontal="Left" vertical="Center"
textPolicy="Wrap"/>
```

```
                </style>
                <title visible="true">
                    <style>
                        <border thickness="None">
                            <color rgb="#000000" alpha="255"/>
                        </border>
                        <background>
                            <color rgb="#000000" alpha="0"/>
                        </background>
                        <font size="8" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false"
                            rgb="#000000"/>
                        <alignment horizontal="Left" vertical="Center"
textPolicy="Truncate"/>
                    </style>
                    <layout orientation="Auto" spacing="2"/>
                </title>
                <layout groupByDimension="false" symbolSize="9" location="Right"
orientation="Auto" spacing="4" adjust="false"/>
                <extraInfo visible="true">
                    <font size="10" face="Arial" italic="false" bold="false"
strikethrough="false" underline="false" rgb="#7070ff"/>
                </extraInfo>
            </legend>
            <plotArea>
                <tagCloudSeries levelColoring="45" comparator="Weight"
mode="Wordle" orientation="HorizontalAndVertical"
                    fillRate="0.65" spacingBetweenTags="Fixed"
spacingValue="0.0">
                    <tag maxFontRatio="Fixed" maxFontSize="8"
minFontRatio="Fixed" minFontSize="4" minVisibleFontSize="4"
                        fontScaling="Logarithmic">
                        <font size="6" face="Arial" italic="false" bold="true"
strikethrough="false" underline="false"
                            rgb="#555555"/>
                    </tag>
                </tagCloudSeries>
            </plotArea>
            <graphics>
                <coloring>
                    <palettes>
                        <palette alpha="50" refId="red" />
                    </palettes>
                    <tagCloudSeries method="CustomRange">
                        <outOfRange>
                            <color rgb="#c0c0c0" alpha="117"/>
                        </outOfRange>
                        <nullOrEmpty>
                            <color rgb="#e0e0e0" alpha="125"/>
                        </nullOrEmpty>
                        <data distributionMode="ByQuantiles"
intervalSyntax="US"/>
                        <ranges percentage="true">
                            <range from="0.0" to="33.0">
                                <color rgb="#ff0000" alpha="255"/>
                            </range>
                            <range from="33.0" to="67.0">
                                <color rgb="#000000" alpha="255"/>
                            </range>
                            <range from="67.0" to="100.0">
                                <color rgb="#00ff00" alpha="255"/>
                            </range>
                        </ranges>
                    </tagCloudSeries>
                </coloring>
                <rendering filter="None">
                    <shadow xOffset="0.0" yOffset="0.0" filterPassCount="3"
filterWindowSize="5" lightPower="0.3">
```

```
                        <color rgb="#9d9d9d" alpha="187"/>
                    </shadow>
                </rendering>
            </graphics>
            <axes>
                <axis role="Category" optional="false">
                    <id>0</id>
                    <name>Tags Name</name>
                    <expressions>
                        <formula dataType="String"
dataObjectId="DP0.DO5">=[Service]</formula>
                    </expressions>
                </axis>
                <axis role="TagsWeight" optional="false">
                    <id>1</id>
                    <name>Tags Weight</name>
                    <expressions>
                        <formula polarity="Auto" dataType="Numeric"
dataObjectId="DP0.DO7">=[Revenue]</formula>
                    </expressions>
                </axis>
                <axis role="TagsFamily" optional="true">
                    <id>2</id>
                    <name>Tags Family</name>
                    <expressions>
                        <formula polarity="Auto" dataType="Numeric"
dataObjectId="DP0.DO25">=[Future guests]</formula>
                    </expressions>
                </axis>
            </axes>
        </chart>
    </content>
</element>
```

Example: Custom Element

Request body:

```
<element type="Custom">
    <parentId>2</parentId>
    <size minimalWidth="9.141" minimalHeight="2.473"/>
    <position x="0.25" y="0.55" oneHorizontalPage="false"
newHorizontalPage="false" repeatOnEveryVerticalPage="false"
oneVerticalPage="false" newVerticalPage="false"/>
    <hide always="false" whenEmpty="false"/>
    <style>
        <border>
            <top thickness="None" rgb="#000000" style="None"/>
            <bottom thickness="None" rgb="#000000" style="Dotted"/>
            <left thickness="None" rgb="#00ee00" style="None"/>
            <right thickness="None" rgb="#000000" style="None"/>
        </border>
    </style>
    <content>
        <custom type="area" serverId="Delegated_Rendering_Server">
            <layout showDimensionsWithEmptyMeasureValues="false"
showDimensionsWithMeasuresEqualToZero="false"
showDimensionsWithSumOfMeasuresEqualToZero="false"
showMeasuresWithEmptyDimensionValues="true" duplicateRowAggregation="false"/>
            <axes>
                <axis role="category-axis" optional="false">
                    <name>Category Axis</name>
                    <expressions>
```

```
                        <formula dataType="String">=[Resort]</formula>
                        <formula dataType="String">=[Region]</formula>
                        <formula dataType="String">=[City]</formula>
                    </expressions>
                </axis>
                <axis role="region-color" optional="true">
                    <name>Color</name>
                    <expressions>
                        <formula dataType="String">=[Year]</formula>
                    </expressions>
                </axis>
                <axis role="primary-values" optional="true">
                    <name>Values</name>
                    <expressions>
                        <formula dataType="Numeric">=[Future guests]</formula>
                    </expressions>
                </axis>
            </axes>
        </custom>
    </content>
</element>
```

Related Information

Charts [page 69]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.9 Inserting a Row or Column into a Table

Usage

Inserts a row or column into a table according to the position of a given cell.

Request

```
POST /documents/<documentID>/reports/<reportID>/elements/<elementID>?
strip=<strip>&position=<position>
```

Where:

- `<strip>` is a string parameter that defines the row or column to insert into the table that contains the given cell `<elementID>`. Values are `row` or `column`.
- `<position>` is a string parameter that defines the position of the row or column relative to the given cell. Values are:
  - `left` or `right` for a column
  - `above` or `below` for a row

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: Inserting a Row Above the Given Cell (JSON)

`POST /documents/16654/reports/1/elements/6?strip=row&position=above`

```
{"success":
    {
     "message":"The resource of type "Report element" with identifier "6" has
been successfully updated.",
     "id":6
    }
}
```

8.4.10 Merging or Splitting Cells of a Table

Usage

Merges cells of a table into one cell or splits the previously merged cells.

Request

`PUT /documents/<documentID>/reports/<reportID>/elements/<elementID>?operation=<operation>&targetCellIds=<targetCellIDs>`

Where:

- `<operation>` is a mandatory parameter of type string that defines the operation to be applied on the cell. Values are `split` or `merge`.
- `<targetCellIDs>` is a mandatory parameter that lists the identifiers of the cells to be split or merged, separated by a comma

Operation Rules

Merge rules:

- The list of identifiers must contain at least two cells.

- The cells in the list of identifiers must be all inside the same cell matrix.
- The cells must form a rectangle zone in the table.

Split rules:

- The cell is inside a cell matrix.
- The cell has the row or column span greater than 1.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: Merging Two Cells of a Table (XML)

`PUT /documents/16654/reports/1/elements/12?operation=merge&targetCellIds=4,6`

```
<success>
    <message>The resource of type "Report element" with identifier "12" has been
successfully updated.</message>
    <id>12</id>
</success>
```

Example: Splitting Three Cells of a Table (JSON)

`PUT /documents/16654/reports/1/elements/12?operation=split&targetCellIds=8,10,9`

```
{"success":
    {"message":"The resource of type "Report element" with identifier "12" has
been successfully updated.",
     "id":12}
}
```

8.4.11 Setting a Table Cell as Section

Usage

Sets a table cell as section.

Request

```
PUT /documents/<documentId>/reports/<reportId>/elements/<elementId>?
operation=setAsSection
```

Where `operation` is a string parameter that indicates the action on the table cell.

Response

Response type: `application/xml` or `application/json`.

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/8644/reports/1/elements/4?operation=setAsSection
```

Response body:

```
<success>
    <message>The resource of type "Report element" with identifier "40" has been
successfully created.</message>
    <id>40</id>
</success>
```

8.4.12 Exporting a Report Element

Usage

Exports a report element in one of the following formats:

- HTML
- Zipped HTML
- MHTML (multipart HTML)
- XML
- PDF
- Microsoft Excel 2003
- Microsoft Excel 2007
- CSV
- Plain text
- Image of type PNG, JPEG, GIF or BMP if the report element is a chart. If the report element is a custom element and its service returns a picture, then the call returns the picture.

> **⚠ Caution**
>
> - You cannot export a chart or custom element to CSV or plain text.
> - You cannot export a whole page zone (header, footer, or body). This generates a 400 HTTP request.

> **ⓘ Note**
>
> - If you choose the HTML format, the web service generates the image links. Therefore, the logon token must still be valid when the HTML output is displayed.
> - Exports to HTML will be optimized for the end-user browser if you use the User-Agent HTTP header in the REST API call.

> **ⓘ Note**
>
> Since 4.2 SP4, if drill mode and drill output mode are both activated for at least one of the reports, then drill information is included in `text/xml` output.

> **ⓘ Note**
>
> Before 4.2 SP4, default value for dpi was set to 300 for PDF and 96 for other output types. Since 4.2 SP4, the default behavior is to keep the server's settings.

> **ⓘ Note**
>
> Before 4.2 SP4, the default value for Excel data optimization was set to false. Since 4.2 SP4, the default behaviour is to keep the server's settings.

Since 4.2 SP4, it is possible to export a custom element as an image (if supported by the custom element server). With `GET image/*`, the rendered type is chosen, among the supported image types, preferable in the following order: PNG, JPEG, GIF, BMP, and so on. Before 4.2 SP4, if it was supported, BMP was returned first.

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>?
<optional_parameters>
```

Optional Parameters

Parameter	Description	Supported Formats
`datapath`	Optional. A string that specifies the data path, for example `{elementId}?datapath=DP1.DObc:"2004",DP1.DOa6:"Austin"`.	All
`reference`	Optional. A string that specifies a reference, for example `{elementId}?reference=1.D.7`.	All

Parameter	Description	Supported Formats
`dpi`	Resolution in dots per inch (dpi) for generated charts. Value between 75 and 9600. Default is 300 for PDF format, 96 for all other formats.	All except XML and CSV
`mode`	`normal` or `quickDisplay`.	All
`orientation`	Page orientation. Use to force a specific page orientation. Values are `portrait` and `landscape`.	If `mode=normal`
`widthScaling`	Number of pages per report displaying in width. Default is `0` and means no constraint in width.	If `mode=normal`
`heightScaling`	Number of pages per report displaying in height. Default is `0` means no constraint in height.	If `mode=normal`
`unit`	A string that defines the unit that sizes will be reported in. Values are `"metric"`, `"millimeter"`, `"point"`, and `"pixel"` (default).	XML
`rawValues`	Boolean. Default is `false`. If `true`, the raw values and their types are exported with the formatted values.	XML
`chartOutputFormat`	Output format for generated chart. Values are: `jpeg`, `bmp`, `gif`, and `png` (default).	HTML, ZIP, and MHTML
`imageUrl`	String used to customize image links when they cannot be reached from the information system.	HTML
`optimized`	Boolean. Default is `false`. If `true`, the generated output is optimized for calculations inside Microsoft Excel.	Microsoft Excel 2003 and Microsoft Excel 2007
`textQualifier`	Character used to surround each column value. Values are `'` or `"`.	CSV
`columnDelimiter`	String that defines a character put between columns. Values are comma (`,`), semi-colon (`;`) or the special string `Tab`.	CSV
`charset`	String that defines a valid server charset retrieved from the list of charsets.	CSV

> **⚠ Caution**
>
> You cannot use `datapath` and `reference` parameters in the same URL. This generates a 400 HTTP request.

Customization of Image Links

You may need to customize the source of an image in a report element exported as HTML if the image is unreachable by your system:

```
<img border="0"
style="position:absolute;top:0px;left:0px;width:400px;height:300px;"
    alt="3D Column Chart" src="http://dewdfwadept1054.dhcp.wdf.sap.corp:6405/
biprws/
    raylight/v1/documents/6406/reports/1/elements/13/images/
dxXMLDraft.drilloff_1*2*6?
    X-SAP-LogonToken={X-SAP-LogonToken}">
</img>
```

To do this, use the `imageUrl` parameter with an appropriate value in the `GET` call:

```
GET /documents/6406/reports/1/elements/13?
reference=1.D.7&imageUrl=http%3A%2F%2FmyServer%2FmyPage.jsp
```

The resulting image link looks like:

```
<img border="0"
style="position:absolute;top:0px;left:0px;width:400px;height:300px;"
    alt="3D Column Chart" src="http://myServer/myPage.jsp?
documentId=6406&reportId=
    1&imageId=dxXMLDraft.drilloff_1*2*6&X-SAP-LogonToken={X-SAP-LogonToken}">
</img>
```

Response

Response type:

- `text/xml` for XML
- `text/html` for HTML
- `application/zip` for zipped HTML
- `multipart/related` for MHTML
- `application/pdf` for PDF
- `application/vnd.ms-excel` for Microsoft Excel 2003
- `application/vnd.openxmlformats-officedocument.spreadsheetml.sheet` for Microsoft Excel 2007
- `text/csv` for CSV
- `text/plain` for plain text
- `image/bmp` for BMP images
- `image/gif` for GIF images
- `image/png` for PNG images
- `image/jpeg` or `image/jpg` for JPEG images
  `image/jpg` is the returned content type in both cases.

Example: HTML

```
curl -G -s -H "accept:text/html" -H X-SAP-LogonToken:"""%tokenValue%"""
 "<base_webi_REST_URL>/documents/9227/reports/1/elements/4" > reportelement4.htm
```

Example: XML

```
curl -G -s -H "accept:text/xml" -H X-SAP-LogonToken:"""%tokenValue%"""
 "<base_webi_REST_URL>/documents/9227/reports/1/elements/4" > reportelement4.xml
```

Example: PDF

```
curl -G -s -H "accept:application/pdf" -H X-SAP-LogonToken:"""%tokenValue%"""
 "<base_webi_REST_URL>/documents/9227/reports/1/elements/4" > reportelement4.pdf
```

Example: Microsoft Excel 2003

```
curl -G -s -H "accept:application/vnd.ms-excel" -H "X-SAP-
LogonToken:""%tokenValue%"""
 "<base_webi_REST_URL>/documents/9227/reports/1/elements/4?
dpi=150&optimized=true" > reportelement4.xls
```

Example: Microsoft Excel 2007

```
curl -G -s -H "accept:application/vnd.openxmlformats-
officedocument.spreadsheetml.sheet"
 -H X-SAP-LogonToken:"""%tokenValue%"""
 "<base_webi_REST_URL>/documents/9227/reports/1/elements/4" >
reportelement4.xxlsx
```

Example: CSV

The column delimiter in this example is the comma.

```
curl -G -s -H "accept:text/csv" -H X-SAP-LogonToken:"""%tokenValue%"""
 "<base_webi_REST_URL>/documents/9227/reports/1/elements/4?
textQualifier='&columnDelimiter=,&charset='UTF-8'" > reportelement4.csv
```

Example: Image

```
curl -G -s -H "accept:image/bmp" -H X-SAP-LogonToken:"""%tokenValue%"""
 "<base_webi_REST_URL>/documents/9227/reports/1/elements/4?
imageUrl=http%3A%2F%2FmyServer%2FmyPage.jsp" > reportelement4.bmp
```

Example: Plain Text

```
GET /documents/6872/reports/1/elements/13
```

```
Accept: text/plain
Content-Type: text/plain
HTTP Response Code: 200
Response Time: 5721 ms
Response:
Austin
    e-Fashion Austin
2004    $561,123
2005    $1,003,071
2006    $1,135,479
Boston
    e-Fashion Boston Newbury
2004    $238,819
...
```

Example: Chart in Plain Text

```
GET /documents/6868/reports/1/elements/13?reference=1.D.7
```

```
Thread ID: 18
Accept: text/plain
Content-Type: application/xml
HTTP Response Code: 406
Response Time: 9805 ms
Response:
<?xml version="1.0" encoding="UTF-8"?>
<error>
    <error_code>WSR 00103</error_code>
    <message>The resource of type "Report element" with identifier "13" is not
compatible with content type "[text/plain]".</message>
</error>
```

Related Information

Getting the List of Documents [page 226]
Getting the Charsets [page 270]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.13 Deleting a Report Element

Usage

You can delete one of the following report elements:

- Cell (only free cells, not cells in a table)
- Section
- Table (VTable, HTable, XTable, and Form)
- Row or column of a table that contains a given cell
- Visualization (charts)
- Custom elements

> → Remember
>
> - The content of that element is also deleted from the report.
> - You cannot delete PageZone elements and table elements of type Cell.

Request

```
DELETE /documents/<documentID>/reports/<reportID>/elements/<elementID>?strip=<strip>
```

Where:

- `<strip>` is an optional, string parameter that defines the row or column to remove from the table that contains the given cell `<elementID>`. Values are `row` or `column`. This parameter is mandatory if the cell is in the table header or footer. It is not necessary if the cell is in the table body, because the corresponding row or column will be deleted automatically depending on the table type.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: Deleting a Table (XML)

```
DELETE /documents/5022/reports/3/elements/4
```

```
<success>
    <message>The resource of type "Report element" with identifier "4" has been
successfully deleted.</message>
    <id>4</id>
```

```
</success>
```

Example: Deleting the Row Containing the Given Cell of a Table Header (JSON)

```
DELETE /documents/5022/reports/3/elements/22?strip=row
```

```
{"success":{
    "message":"The resource of type "Report element" with identifier "22" has
been successfully deleted.",
    "id":22
    }
}
```

Example: Deleting the Row Containing the Given Cell of a Table Body (XML)

```
DELETE /documents/5022/reports/3/elements/23
```

```
<success>
    <message>The resource of type "Report element" with identifier "23" has been
successfully deleted.</message>
    <id>23</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.14 Ordering a Report Element

Usage

Changes the position of a report element's layer.

Request

```
PUT /documents/<documentId>/reports/<reportId>/elements/<elementId>?operation=order
```

Where:

- `operation` is string a parameter that indicates the action on the report element.
- `level` is an optional string parameter that indicates the report element's layer position. Possible values are `front`, `back`, `forward` and `backward`.

> **Note**
>
> If `level` is set `forward` or `backward`, the report element moves one layer. If set to `front` or `back`, the report element moves on top or underneath of all layers.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/8644/reports/1/elements/12?operation=order&level=front
```

Response body:

```
<success>
    <message>The resource of type "Report element" with identifier "12" has been
updated.</message>
    <id>12</id>
</success>
```

8.4.15 Working with Axes

Below are the tasks you can perform on the axes of a report element:

- Updating or removing the formulas on axes
- Using breaks on table axes to organize the data according to a selected dimension or measure
- Sorting data on the table axes

The following table shows that the number of axes depends on the type of report element.

Report Element	Axis Number
Section	One axis with identifier `0` that represents a row
HTable or Form	One axis with identifier `0` that represents a row
VTable	One axis with identifier `1` that represents a column
XTable	3 axes with identifiers `0` for the row, `1` for the column, and `2` for the body
Visualization	Different axes depending on the type of chart. See Roles and Identifiers [page 101].

Getting the Details of an Axis in a Report Element [page 441]

Updating the Axis of a Report Element [page 443]

Getting the Axis Expressions of a Report Element [page 445]

Updating the Expressions of an Axis [page 446]

Removing the Expressions From an Axis [page 447]

Creating a Table Break [page 448]

Updating a Table Break [page 450]

Deleting a Table Break [page 451]

Deleting all the Breaks in a Report Element [page 452]

Updating all the Breaks on a Table's Axis [page 453]

Listing all the Breaks on a Table's Axis [page 454]

Deleting all the Breaks on a Table's Axis [page 455]

Getting the Details of a Break in a Table [page 456]

Updating the Details of a Report Element Axis Break [page 457]

Updating the Sortings of an Axis (DEPRECATED since 4.2 SP04) [page 458]

Removing the Sortings From an Axis (DEPRECATED since 4.2 SP04) [page 459]

Getting the Axis Sorts in a Report Element [page 460]

Getting the Details of a Break on a Table's Axis [page 462]

8.4.15.1 Getting the Details of an Axis in a Report Element

Usage

Use this URL to get the details of an axis in a report element (Section, Table, Chart or Custom Element).

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>
```

Parameters

Path parameters

- `<documentID>`: document identifier retrieved from the list of documents
- `<reportID>`: identifier of the document report
- `<elementID>`: identifier of the report element
- `<axisID>`: identifier of the element axis

Optional

- `<datapath>`: (type=string) specify the datapath of the report element instance
- `<reference>`:(type=string) identifier of a report element instance

> **Note**
>
> You can't use the `datapath` and `reference` parameters at the same time.

Response

Response type: `application/xml` or `application/json`

Example: Get the axis of a report element (all instances) (XML Format)

```
[GET] /documents/11922/reports/1/elements/36/axes/1
```

Response:

```
<axis role="Column" fold="false">
    <id>1</id>
    <expressions>
        <formula dataType="String" dataObjectId="DP0.DO5">=[Service]</formula>
        <formula dataType="Numeric" dataObjectId="DP0.DO14">=[Number of guests]</
formula>
    </expressions>
</axis>
```

Example: Get the axis of a specific instance of a report element (XML Format)

```
[GET] /documents/11922/reports/1/elements/36/axes/1?
datapath=DP0.DObc%3A%222005%22%2CDP0.DOda%3A%22Florida%22
```

Response:

```
<axis role="Column" fold="true">
    <id>1</id>
    <expressions>
        <formula dataType="String" dataObjectId="DP0.DO5">=[Service]</formula>
        <formula dataType="Numeric" dataObjectId="DP0.DO14">=[Number of guests]</
formula>
    </expressions>
</axis>
```

> **Note**
>
> Note: The only information related to a specific instance of a report element is the "fold" attribute.

8.4.15.2 Updating the Axis of a Report Element

Usage

Use this URL to update an axis in a report element (Section, Table, Visualization or Custom).

Request

`PUT /documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>`

Request body: the break to modify.

Parameters

Required

- `<documentID>`: (type=integer) a valid identifier of a Web Intelligence document.
- `<reportID>`: (type=integer) a valid identifier of a report inside the Web Intelligence document.
- `<elementId>`: (type=integer) a valid element identifier.
- `<axisId>`: (type=integer) a valid axis identifier.

Optional

- `<datapath>`: (type=string) specify the datapath of the report element instance
- `<reference>`: (type=string) an identifier of a report element instance

> **Note**
>
> You can't use the `datapath` and `reference` parameters at the same time.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: Update an axis of a report element (all instances)

```
[PUT] <url>/documents/21953/reports/658/elements/659/axes/1
```

Request body:

```
<axis role="primary-values">
    <expressions>
        <formula dataType="Numeric">=[Number of guests]</formula>
    </expressions>
</axis>
```

Response:

```
<success>
    <message>The resource of type "Axis" with identifier "1" has been
successfully updated.</message>
    <id>0</id>
</success>
```

Example: Update an axis of a specific instance of a report element

```
[PUT] <url>/documents/11931/reports/48/elements/75/axes/1?
datapath=DP0.DObc%3A%222005%22%2CDP0.DOda%3A%22Florida%22
```

Request body:

```
<axis role="Column" fold="true">
    <id>1</id>
    <expressions>
        <formula dataType="String" dataObjectId="DP0.DO5">=[Service]</formula>
        <formula dataType="Numeric" dataObjectId="DP0.DO14">=[Number of guests]</
formula>
    </expressions>
</axis>
```

> **Note**
>
> The only information related to a specific instance of a report element is the "fold" attribute.

Response:

```
<success>
    <message>The resource of type "Axis" with identifier "1" has been
successfully updated.</message>
    <id>0</id>
```

```
</success>
```

8.4.15.3 Getting the Axis Expressions of a Report Element

Usage

Get the expressions of a report element (Table, Section, Chart, and Custom Element). Expressions define which dimensions feed the axis.

Request

`GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/expressions`

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<reportID>`: identifier of the document report
- `<elementID>`: identifier of the report element
- `<axisID>`: identifier of the element axis

Response

Response type: `application/xml` or `application/json`

Example: XML Format

`[GET] <url>/documents/16995/reports/1/elements/8/axes/0/expressions`

Response:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<expressions>
    <formula dataType="String">=[Country]</formula>
</expressions>
```

8.4.15.4 Updating the Expressions of an Axis

Usage

Updates the expressions of an axis in a report element of one of the following types:

- Section
- Table (HTable, VTable, XTable, and Form)
- Visualization (charts)
- Custom elements

Request

```
PUT /documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/
expressions
```

Request body: the expressions to update.

> **Restriction**
>
> You can update the expressions of standard tables only, otherwise the table formatting will be lost. This feature corresponds to the *Assign Data* contextual menu used to create tables in SAP BusinessObjects Web Intelligence.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/16995/reports/1/elements/8/axes/0/expressions
```

Request body:

```
<expressions>
    <formula dataType="String">=[Resort]</formula>
    <formula dataType="Numeric">=[Revenue]</formula>
</expressions
```

Response:

```
<success>
```

```
    <message>The resource of type "Axis" with identifier "0" has been
successfully updated.</message>
    <id>0</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.15.5 Removing the Expressions From an Axis

Usage

Deletes the expressions of an axis in a report element of one of the following types:

- Section
- Table (HTable, VTable, XTable, and Form)
- Visualization (charts)
- Custom elements

Request

```
DELETE /documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/
<axisID>/expressions
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/17170/reports/3/elements/8/axes/1/expressions
```

```
<success>
```

```
    <message>The resource of type "Axis" with identifier "1" has been
successfully updated.</message>
    <id>1</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.15.6 Creating a Table Break

Usage

Creates a break in the specified axis of a report element of type table.

> → Remember
>
> You cannot create breaks in the following cases:
>
> - In a Form table
> - In the body of an XTable

Request

```
POST /documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/
breaks
```

Request body: the details of the break.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

Creating a simple break

```
POST /documents/17230/reports/3/elements/8/axes/0/breaks
```

Request body:

```
<break onePage="true" newPage="true" sort="false" duplicate="Repeat"
repeatHeader="true" showHeader="true" showFooter="false">
    <formula dataType="String">=[Year]</formula>
</break>
```

Response:

```
<success>
    <message>The resource of type "Break" with identifier "1" has been
successfully created.</message>
    <id>1</id>
</success>
```

Example

Creating a same-level break

```
POST /document/17230/reports/3/elements/9/axes/0/breaks
```

Request body:

```
<break onePage="false" newPage="false" sort="true" duplicate="Remove"
repeatHeader="false" showHeader="true" showFooter="true">
    <id>1</id>
    <formula dataType="String" dataObjectId="DP1.DO84"
qualification="Dimension">=[Query 2].[Country]</formula>
    <formula dataType="String" dataObjectId="DP1.DO12"
qualification="Dimension">=[Query 2].[Region]</formula>
    <formula dataType="String" dataObjectId="DP1.DO13"
qualification="Dimension">=[City]</formula>
</break>
```

Example

Creating a value-based break

```
POST /document/17230/reports/3/elements/7/axes/0/breaks
```

```
</breaks>
    <break onePage="false" newPage="false" sort="true" duplicate="Remove"
       repeatHeader="false" showHeader="false" showFooter="true">
    <id>2</id>
    <formula dataType="String" dataObjectId="DP0.DO3"
qualification="Hierarchy">=[COUNTRY_01]</formula>
    <values>
     <value id="0000000011">Level01\ 11 - Clean Air Tran</value>
```

```
     <value id="0000000032">Level01\Level02\Level03\Level04\ 32 - Spokes</value>
     <value id="0000000090">Level01\Level02\Level03\Level04\Level05\ 90 - Sports
Alsace</value>
     <value id="0000000001">Level01\Level02\Level03\Level04\Level05\ 1 - City
Cyclists</value>
    </values>
    </break>
</breaks>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.15.7 Updating a Table Break

Usage

Modifies the content of break of an axis in a report element of type table.

Request

```
PUT /documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/
breaks/<breakID>
```

Request body: the break to modify.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/17230/reports/3/elements/8/axes/0/breaks/1
```

Request body:

```
<break onePage="true" newPage="true" sort="false" duplicate="Repeat"
repeatHeader="true" showHeader="true" showFooter="false">
    <formula dataType="String">=[Year]</formula>
</break>
```

Response:

```
<success>
    <message>The resource of type "Break" with identifier "1" has been
successfully updated.</message>
    <id>1</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]
Getting the Details of a Report Element [page 400]

8.4.15.8 Deleting a Table Break

Usage

Removes a specific break of an axis in a report element of type table.

Request

```
DELETE /documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/
<axisID>/breaks/<breakID>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/17230/reports/3/elements/8/axes/0/breaks/1
```

Response:

```
<success>
    <message>The resource of type "Break" with identifier "1" has been
successfully removed.</message>
    <id>1</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]
Getting the Details of a Report Element [page 400]

8.4.15.9 Deleting all the Breaks in a Report Element

Usage

Deletes all the breaks in a table specified in the URL parameter.

Request

```
DELETE /documents/<documentId>/reports/<reportId>/elements/<elementId>/breaks
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example (XML)

```
DELETE /documents/17230/reports/3/elements/8/breaks
```

Response body:

```
<success>
    <message>The resource of type "Break" has been successfully removed.</
message>
    <id>0</id>
</success>
```

8.4.15.10 Updating all the Breaks on a Table's Axis

Usage

Updates all breaks on a table's axis specified in the URL parameter.

Request

```
PUT /documents/<documentId>/reports/<reportId>/elements/<elementId>/axes/<axisId>/
breaks
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example (XML)

```
PUT /documents/22115/reports/44/elements/61/axes/1/breaks
```

Request body:

```
<breaks>
    <break>
        <formula>=[Service Line]</formula>
        <values>
            <value>Recreation</value>
```

```
            <value>Travels</value>
        </values>
    </break>
    <break>
        <formula>=[Resort]</formula>
    </break>
    <break>
        <formula>=[City]</formula>
        <formula>=[Region]</formula>
    </break>
    <break>
        <formula>=[Service]</formula>
    </break>
</breaks>
```

Response body:

```
<success>
    <message>The resource of type "Axis" with identifier "1" has been
successfully updated.</message>
    <id>1</id>
</success>
```

8.4.15.11 Listing all the Breaks on a Table's Axis

Usage

This URL lists all the breaks on a table's axis specified in the URL parameter.

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/
breaks
```

Response

Response type: `application/xml` or `application/json`

Example: XML Format:

```
[GET] <url>/documents/11014/reports/31/elements/36/axes/1/breaks
```

Response:

```
<breaks>
    <break onePage="false" newPage="false" sort="true" duplicate="Remove"
repeatHeader="false" showHeader="false" showFooter="true">
        <id>1</id>
        <formula dataType="String" dataObjectId="DP0.DO39">=[Country]</formula>
    </break>
    <break onePage="false" newPage="false" sort="true" duplicate="Remove"
repeatHeader="false" showHeader="true" showFooter="true">
        <id>2</id>
        <parentId>1</parentId>
        <formula dataType="String" dataObjectId="DP0.DO31">=[Year]</formula>
    </break>
</breaks>
```

8.4.15.12 Deleting all the Breaks on a Table's Axis

Usage

Deletes all the breaks on a table's axis specified in the URL parameter.

Request

```
DELETE /documents/<documentId>/reports/<reportId>/elements/<elementId>/axes/
<axisId>/breaks
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request

Example (XML)

```
DELETE /documents/17230/reports/3/elements/8/axes/0/breaks
```

Response body:

```
<success>
    <message>The resource of type "Axis" with identifier "0" has been
successfully updated.</message>
```

```
    <id>0</id>
</success>
```

8.4.15.13 Getting the Details of a Break in a Table

Usage

This URL returns the details of the breaks on a table's axis specified in the URL parameter.

Request

`GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/breaks/<breakID>`

Parameters

Parameter	Description
`<documentID>`	A valid identifier of a Web Intelligence document.
`<reportID>`	A valid identifier of a report of the Web Intelligence document..
`<elementID>`	A valid identifier of a report element
`<axisID>`	A valid identifier of an axis A Section has only one axis with identifier 0 (role 'Row'). Tables have the following identifiers: • 'HTable' or 'Form' has only one axis with identifier 0 (role 'Row') • 'VTable' one axis with identifier 1 (role 'Column') • 'XTable' has 3 axes with identifiers 0 (role 'Row'), 1 (role 'Column') and 2 (role 'Body'). Visualizations may have different axes depending of the chart type. For more information, please refer to Charts.
`<breakID>`	A valid identifier of a break.

Response

Response type: `application/xml` or `application/json`

Example: XML Format: All table's instances

```
[GET] <url>/documents/11026/reports/31/elements/36/axes/0/breaks/2
```

Response:

```
<break onePage="false" newPage="false" sort="true" duplicate="Remove"
repeatHeader="false" showHeader="true" showFooter="true" fold="false">
    <id>2</id>
    <parentId>1</parentId>
    <formula dataType="String" dataObjectId="DP0.DO31">=[Year]</formula>
</break>
```

> **Note**
>
> The attribute 'fold' reflects in this case if all the instances of the break are folded.

Example: XML Format: A specific instance of the table

```
[GET] <url>/documents/11949/reports/1/elements/17/axes/1/breaks/1?
datapath=DP0.DObc%3A%222004%22%2CDP0.DOda%3A%22Texas+%22
```

Response:

```
<break onePage="false" newPage="false" sort="true" duplicate="Repeat"
repeatHeader="false" showHeader="false" showFooter="true" fold="true">
    <id>1</id>
    <formula dataType="String" dataObjectId="DP0.DOda">=[State]</formula>
</break>
```

> **Note**
>
> The attribute 'fold' reflects in this case if this specific instance of the break is folded.

8.4.15.14 Updating the Details of a Report Element Axis Break

Usage

Use this URL to edit the details of a report element axis break.

Request

```
PUT /documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/
breaks/<breakID>
```

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<reportID>`: identifier of the document report
- `<elementID>`: identifier of the report element
- `<axisID>`: identifier of the element axis
- `<breakID>`: identifier of the break

Optional parameters

- `<datapath>`: (type=string) allows to specify the datapath of the table's instance
- `<reference>`: (type=string) identifier of a report element instance.

> **Note**
>
> You can't use the `datapath` and `reference` parameters at the same time.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

8.4.15.15 Updating the Sortings of an Axis (DEPRECATED since 4.2 SP04)

Usage

To update the sortings of an axis in a report element of type Table (HTable, VTable, XTable, and Form), Section, Visualization, or Custom Element see Updating a Sort in a Report Element [page 489].

> **Note**
>
> This API is deprecated and should not be used from 4.2 SP04 onwards.

Request

```
PUT /documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/
sorts
```

Request body: the sortings to update.

→ Remember

The sortings will be applied in the specified order.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`PUT /documents/17196/reports/3/elements/12/axes/1/sorts`

Request body:

```
<sorts>
    <sort order="Ascending" dataType="String">=[Resort]</sort>
    <sort order="Ascending" dataType="String">=[Country]</sort>
 </sorts>
```

Response:

```
<success>
    <message>The resource of type "Axis" with identifier "1" has been
successfully updated.</message>
    <id>1</id>
</success>
```

8.4.15.16 Removing the Sortings From an Axis (DEPRECATED since 4.2 SP04)

Usage

To delete the sortings of an axis in a report element of type Table (HTable, VTable, XTable, and Form), Section, Visualization, or Custom Element, see Removing a specific sort of a report element [page 492].

ⓘ Note

This API is deprecated and should not be used from 4.2 SP04 onwards.

Request

```
DELETE /documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/
<axisID>/sorts
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/18930/reports/2/elements/15/axes/1/sorts
```

```
<success>
    <message>The resource of type "Axis" with identifier "1" has been
successfully updated.</message>
    <id>1</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.15.17 Getting the Axis Sorts in a Report Element

Usage

Gets the sort type (ascending or descending) of an axis in a report element of type Table (HTable, VTable, XTable, and Form), Section, Visualization, or Custom Element. This RESTful call allows to get the sort types on an axis of a report element. The supported report elements having axes are:

- Sections (since 4.1 SP6),
- Tables
- Visualizations (since 4.1 SP5)
- Custom (since 4.2 SP3)

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/
sorts
```

Parameters

Parameter	Description
`<documentID>`	A valid identifier of a Web Intelligence document.
`<reportID>`	A valid identifier of a report of the Web Intelligence document..
`<elementID>`	A valid identifier of a report element
`<axisID>`	A valid identifier of an axis A Section has only one axis with identifier 0 (role 'Row'). Tables have the following identifiers: • 'HTable' or 'Form' has only one axis with identifier 0 (role 'Row') • 'VTable' one axis with identifier 1 (role 'Column') • 'XTable' has 3 axes with identifiers 0 (role 'Row'), 1 (role 'Column') and 2 (role 'Body'). Visualizations may have different axes depending of the chart type. For more information, please refer to Charts.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: Get the sorts of a section

```
GET /documents/16995/reports/1/elements/8/axes/0/sorts
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<sorts>
    <sort order="Ascending" dataType="String">=[Resort]</sort>
</sorts>
```

Example: Get the sorts of a table's axis

```
GET /documents/17196/reports/3/elements/12/axes/1/sorts
```

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

```
<sorts>
    <sort order="Ascending" dataType="String">=[Resort]</sort>
    <sort order="Ascending" dataType="String">=[Country]</sort>
</sorts>
```

Example: Get the sorts of a chart's axis

```
GET /documents/13644/reports/6/elements/24/axes/5/sorts
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<sorts>
    <sort order="Descending" dataType="String">=[Service]</sort>
    <sort order="Ascending" dataType="String">=[Country]</sort>
</sorts>
```

Example: Get the sorts of a custom element's axis

```
GET /documents/13644/reports/6/elements/24/axes/1/sorts
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<sorts>
    <sort order="Descending" dataType="String">=[Year]</sort>
</sorts>
```

8.4.15.18 Getting the Details of a Break on a Table's Axis

Usage

This URL gets the details of a break on a table's axis specified in the URL parameter.

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/axes/<axisID>/
breaks/<breakID>
```

Path parameters:

- `<documentID>`: document identifier retrieved from the list of documents
- `<reportID>`: identifier of the document report
- `<elementID>`: identifier of the report element
- `<axisID>`: identifier of the element axis
- `<breakID>`: identifier of the break

Optional parameters

- `<datapath>`: (type=string) allows to specify the datapath of the table's instance
- `<reference>`: (type=string) identifier of a report element instance.

> **Note**
>
> You can't use the `datapath` and `reference` parameters at the same time.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: Getting all a table's instances

```
[GET] <url>/documents/11026/reports/31/elements/36/axes/0/breaks/2
```

Response:

```
<break onePage="false" newPage="false" sort="true" duplicate="Remove"
repeatHeader="false" showHeader="true" showFooter="true" fold="false">
    <id>2</id>
    <parentId>1</parentId>
    <formula dataType="String" dataObjectId="DP0.DO31">=[Year]</formula>
</break>
```

> **Note**
>
> The attribute 'fold' reflects in this case if all the instances of the break are folded.

Example: Getting a specific instance of a table

```
[GET] <url>/documents/11949/reports/1/elements/17/axes/1/breaks/1?
datapath=DP0.DObc%3A%222004%22%2CDP0.DOda%3A%22Texas+%22
```

Response:

```
<break onePage="false" newPage="false" sort="true" duplicate="Repeat"
repeatHeader="false" showHeader="false" showFooter="true" fold="true">
    <id>1</id>
    <formula dataType="String" dataObjectId="DP0.DOda">=[State]</formula>
</break>
```

> **Note**
>
> The attribute 'fold' reflects in this case if this specific instance of the break is folded.

8.4.16 Working with Rankings

Ranking allows you to isolate the top and bottom records in a data set based on a variety of criteria.

Creating a Ranking in a Report Element [page 464]

Updating the Ranking of a Report Element [page 465]

Deleting the Ranking of a Report Element [page 466]

Getting the Ranking in a Report Element [page 467]

8.4.16.1 Creating a Ranking in a Report Element

Usage

Creates a ranking in a report element of type Table (HTable, VTable, XTable, and Form), Section, Visualization, or Custom Element.

> → Remember
>
> You can define only one ranking in a report element.

Request

`POST /documents/<documentID>/reports/<reportID>/elements/<elementID>/ranking`

Request body: the ranking definition.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`POST /documents/17281/reports/1/elements/20/ranking`

Request body:

```
<ranking calculation="Count" top="3" bottom="3">
    <basedOn>=[Number of guests]</basedOn>
```

```
    <rankedBy>=[Year]</rankedBy>
</ranking>
```

Response:

```
<success>
    <message>The resource of type "Ranking" has been successfully created.</
message>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.16.2 Updating the Ranking of a Report Element

Usage

Modifies a ranking in a report element of type Table (HTable, VTable, XTable, and Form), Section, Visualization, or Custom Element.

Request

`PUT /documents/<documentID>/reports/<reportID>/elements/<elementID>/ranking`

Request body: the details of the ranking to update.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`PUT /documents/17281/reports/1/elements/20/ranking`

Request body:

```
<ranking calculation="Count" top="2" bottom="4">
    <basedOn>=[Number of guests]</basedOn>
    <rankedBy>=[Year]</rankedBy>
</ranking>
```

Response:

```
<success>
    <message>The resource of type "Ranking" has been successfully updated.</
message>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.16.3 Deleting the Ranking of a Report Element

Usage

Deletes the ranking in a report element of type Table (HTable, VTable, XTable, and Form), Section, Visualization, or Custom Element.

Request

```
DELETE /documents/<documentID>/reports/<reportID>/elements/<elementID>/ranking
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/17281/reports/1/elements/20/ranking
```

Response:

```
<success>
    <message>The resource of type "Ranking" has been successfully removed.</
message>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.16.4 Getting the Ranking in a Report Element

Usage

This RESTful call allows to get a ranking in a report element of type Table ('VTable', 'HTable', 'XTable', 'Form'), 'Section', 'Visualization' or Custom. Only one ranking can be defined in a report element.

This URL gets a ranking specified in the URL parameter.

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/ranking
```

Response

Response type: `application/xml` or `application/json`

Example: (XML format):

```
[GET] <url>/documents/17281/reports/1/elements/20/ranking
```

Response:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ranking calculation="Count" top="3" bottom="3">
    <basedOn>=[Number of guests]</basedOn>
    <rankedBy>=[Year]</rankedBy>
</ranking>
```

8.4.17 Working with Calculations

Below are the tasks you can perform with calculations of a report element:

- Get the calculations of a report element.
- Create, get details or delete a specific calculation associated to a report element.

Calculations can only apply to cell tables:

- Vertical and horizontal tables: only on body cells
- Cross tables: only on body or header-body cells

Calculation IDs are `Average`, `Count`, `Max`, `Min`, `Percent` and `Sum`. For cells that are based on a non-measure expression, only `Count`, `Max` and `Min` are available.

8.4.17.1 Getting the Calculations of a Report Element

Usage

Returns all the calculations of a report element. The report element can be a table or a table cell.

Request

```
GET /documents/<documentId>/reports/<reportId>/elements/<elementId>/calculations
```

Response

Response type: `application/xml` or `application/json`

Example (XML)

Getting the calculations of a table

```
GET /documents/12236/reports/22/elements/29/calculations
```

Response body:

```
<calculations>
    <calculation horizontal="false">
            <id>Sum</id>
            <formula dataType="Numeric" qualification="Measure">=Sum([Sales
revenue])</formula>
        <basedOn>
            <id>24</id>
            <formula dataType="Numeric" dataObjectId="DP0.DO93"
qualification="Measure">=[Sales revenue]</formula>
        </basedOn>
    </calculation>
    <calculation horizontal="false">
            <id>Count</id>
            <formula dataType="Numeric"
qualification="Measure">=Count([Margin])</formula>
        <basedOn>
            <id>30</id>
            <formula dataType="Numeric" dataObjectId="DP0.DO92"
qualification="Measure">=[Margin]</formula>
        </basedOn>
    </calculation>
    <calculation horizontal="false">
            <id>Average</id>
            <formula dataType="Numeric" qualification="Measure">=Average([Sales
revenue])</formula>
        <basedOn>
            <id>24</id>
            <formula dataType="Numeric" dataObjectId="DP0.DO93"
qualification="Measure">=[Sales revenue]</formula>
        </basedOn>
    </calculation>
    <calculation horizontal="false">
            <id>Max</id>
            <formula dataType="Numeric" qualification="Measure">=Max([Margin])</
formula>
        <basedOn>
            <id>30</id>
            <formula dataType="Numeric" dataObjectId="DP0.DO92"
qualification="Measure">=[Margin]</formula>
        </basedOn>
    </calculation>
    <calculation horizontal="false">
            <id>Percent</id>
            <formula dataType="Numeric"
qualification="Measure">=Percentage([Margin])</formula>
        <basedOn>
            <id>30</id>
            <formula dataType="Numeric" dataObjectId="DP0.DO92"
qualification="Measure">=[Margin]</formula>
        </basedOn>
    </calculation>
</calculations>
```

Getting the calculations of a table cell

```
GET /documents/12236/reports/22/elements/30/calculations
```

Response body:

```
<calculations>
    <calculation horizontal="false">
        <id>Max</id>
        <formula dataType="Numeric" qualification="Measure">=Max([Margin])</
formula>
        <basedOn>
            <id>30</id>
            <formula dataType="Numeric" dataObjectId="DP0.DO92"
qualification="Measure">=[Margin]</formula>
        </basedOn>
    </calculation>
    <calculation horizontal="false">
        <id>Percent</id>
        <formula dataType="Numeric"
qualification="Measure">=Percentage([Margin])</formula>
        <basedOn>
            <id>30</id>
            <formula dataType="Numeric" dataObjectId="DP0.DO92"
qualification="Measure">=[Margin]</formula>
        </basedOn>
    </calculation>
    <calculation horizontal="false">
        <id>Count</id>
        <formula dataType="Numeric" qualification="Measure">=Count([Margin])</
formula>
        <basedOn>
            <id>30</id>
            <formula dataType="Numeric" dataObjectId="DP0.DO92"
qualification="Measure">=[Margin]</formula>
        </basedOn>
    </calculation>
</calculations>
```

8.4.17.2 Getting the Details of a Calculation

Usage

Returns all the information about the specified calculation. The report element can only be a table cell.

Request

```
GET /documents/<documentId>/reports/<reportId>/elements/<elementId>/calculations/
<calculationId>?strip=column
```

Where:

- `strip` is an optional string parameter that defines the direction of the calculation. Possible values are `row` or `column`. In cross tables, `strip` only applies to body cells.

Response

Response type: `application/xml` or `application/json`

Example (XML)

```
GET /documents/15216/reports/5/elements/13/calculations/Sum?strip=row
```

Response body:

```
<calculation horizontal="true">
    <id>Sum</id>
    <formula dataType="Numeric" qualification="Measure">=Sum([Future guests])</
formula>
    <basedOn>
        <id>13</id>
        <formula dataType="Numeric" dataObjectId="DP0.DO25"
qualification="Measure">=[Future guests]</formula>
    </basedOn>
</calculation>
```

8.4.17.3 Creating a Calculation

Usage

Creates a calculation associated to a report element. The report element can be a table or a table cell.

Request

```
POST /documents/<documentId>/reports/<reportId>/elements/<elementId>/calculations?
strip=column
```

Where:

- `strip` is an optional string parameter that defines the direction of the calculation. Possible values are `row` or `column`. In cross tables, `strip` only applies to body cells.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example (XML)

Creating a calculation on a cross table body cell

```
POST /documents/15231/reports/5/elements/13/calculations?strip=row
```

Request body:

```
<calculation>
    <id>Sum</id>
</calculation>
```

Response body:

```
<success>
    <message>The resource of type "Calculation" with identifier "Sum" has been
successfully created.</message>
    <id>Sum</id>
</success>
```

Creating a calculation on a vertical table

```
POST /documents/15246/reports/22/elements/29/calculations
```

Request body:

```
<calculation>
    <id>Count</id>
    <basedOn>
        <id>30</id>
    </basedOn>
</calculation>
```

Response body:

```
<success>
    <message>The resource of type "Calculation" with identifier "Count" has been
successfully created.</message>
    <id>Count</id>
</success>
```

8.4.17.4 Deleting a Calculation

Usage

Deletes a specific calculation associated to a report element. The report element can only be a table cell.

Request

```
DELETE /documents/<documentId>/reports/<reportId>/elements/<elementId>/
calculations/<calculationId>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example (XML)

```
DELETE /documents/15261/reports/22/elements/30/calculations/Max
```

Response body:

```
<success>
    <message>The resource of type "Calculation" with identifier "Max" has been
successfully removed.</message>
    <id>Max</id>
</success>
```

8.4.18 Working with Custom Properties

Getting the List of Custom Properties [page 473]

Getting the Custom Property Value [page 474]

Adding a Custom Property to a Report Element [page 475]

Updating a Custom Property [page 476]

Deleting a Custom Property [page 477]

8.4.18.1 Getting the List of Custom Properties

Usage

Lists all available custom properties of the specified report element.

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/properties
```

Response

Response type: `application/xml` or `application/json`

Response body: the list of keys and values of custom properties.

Example

```
GET /documents/1234/reports/1/elements/100/properties
```

Response:

```
<properties>
    <property key="KEY01">Value01</property>
    <property key="KEY#2">2nd properties</property>
</properties>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.18.2 Getting the Custom Property Value

Usage

Gets the value of the specified custom property of a report element.

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/properties/
<propertyKey>
```

Response

Response type: `application/xml` or `application/json`

Example

`GET /documents/1234/reports/1/elements/100/properties/NEW_KEY`

Response:

```
<property key='NEW_KEY'>MyKeyValue</property>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.18.3 Adding a Custom Property to a Report Element

Usage

Adds a custom property to a report element.

Request

`POST /documents/<documentID>/reports/<reportID>/elements/<elementID>/properties`

Request body: the property to add.

> → Remember
>
> The property key must not be null or empty.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
POST /documents/1234/reports/1/elements/100/properties
```

Request body:

```
<property key='NEW_KEY'>MyKeyValue</property>
```

Response:

```
<success>
    <message>The resource of type "Property" has been successfully created.</
message>
    <id>NEW_KEY</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.18.4 Updating a Custom Property

Usage

Changes the value of the specified custom property of a report element.

Request

```
PUT /documents/<documentID>/reports/<reportID>/elements/<elementID>/properties
```

Request body: the custom property to edit (key and value).

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/1234/reports/1/elements/100/properties
```

Request body:

```
<property key='NEW_KEY'>Modified Value</property>
```

Response:

```
<success>
    <message>The resource of type "Property" has been successfully updated.</
message>
    <id>NEW_KEY</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.18.5 Deleting a Custom Property

Usage

Deletes a custom property from a report element.

Request

```
DELETE /documents/<documentID>/reports/<reportID>/elements/<elementID>/properties/
<propertyKey>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/1234/reports/1/elements/100/properties/MYKEY
```

```
<success>
    <message>The resource of type "Property" has been successfully removed.</
message>
    <id>MYKEY</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.19 Managing Data Filters in Report Elements

APIs are provided to perform tasks on the data filters that apply to a report element.

Creating a Data Filter for a Report Element [page 478]

Getting the Details of a Data Filter for a Report Element [page 480]

Updating the Data Filter for a Report Element [page 481]

Deleting a Data Filter from a Report Element [page 483]

8.4.19.1 Creating a Data Filter for a Report Element

Usage

Creates a data filter for a specific report element, referenced by its ID.

Request

```
POST /documents/<documentID>/reports/<reportID>/elements/<elementID>/datafilter
```

Request body: the description of the data filter to create for the report element. The `key` attribute should be one of the `formulaLanguageId` elements of the report.

The operator is one of the available report filters:

- `Equal`
- `NotEqual`
- `Greater`
- `GreaterOrEqual`
- `Less`
- `LessOrEqual`
- `Between`
- `NotBetween`
- `InList`
- `NotInList`
- `IsNull`
- `IsNotNull`
- `IsAny`
- `Like`
- `NotLike`
- `Both`
- `Except`

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

Request body:

```
<datafilter>
    <and>
        <condition key="[Lines]" operator="InList">
            <value>City Trousers</value>
            <value>Shirt Waist</value>
            <value>Jackets</value>
       </condition>
       <condition key="[State]" operator="NotEqual">
           <value>Colorado</value>
       </condition>
        <or>
            <condition key="[Sales revenue]" operator="Greater">
                <value>5000</value>
            </condition>
            <condition key="[Sales revenue]" operator="Less">
```

```
                <value>10</value>
            </condition>
        </or>
    </and>
</datafilter>
```

Response:

```
<success>
    <message>The resource of type "Report" with identifier "1" has been
successfully updated.</message>
    <id>1</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.19.2 Getting the Details of a Data Filter for a Report Element

Usage

Gets the details of a report element data filter from a report.

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/datafilter
```

Response

Response type: `application/xml` or `application/json`

Example

```
GET /documents/4567/reports/1/elements/22/datafilter
```

Response:

```
<datafilter>
    <and>
        <condition key="[Lines]" operator="InList">
            <value>City Trousers</value>
            <value>Shirt Waist</value>
            <value>Jackets</value>
        </condition>
        <condition key="[State]" operator="NotEqual">
            <value>Colorado</value>
        </condition>
        <or>
            <condition key="[Sales revenue]" operator="Greater">
                <value>5000</value>
            </condition>
            <condition key="[Sales revenue]" operator="Less">
                <value>10</value>
            </condition>
        </or>
    </and>
</datafilter>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.19.3 Updating the Data Filter for a Report Element

Usage

Updates the data filter applied to a given report specified in the URL parameter.

Request

`PUT /documents/<documentID>/reports/<reportID>/elements/<elementID>/datafilter`

Request body: the description of the data filter to create for the report. The operator is one of the available report filter conditions :

- `Equal`
- `NotEqual`
- `Greater`
- `GreaterOrEqual`

- `Less`
- `LessOrEqual`
- `Between`
- `NotBetween`
- `InList`
- `NotInList`
- `IsNull`
- `IsNotNull`
- `IsAny`
- `Like`
- `NotLike`
- `Both`
- `Except`

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`PUT /documents/223/reports/1/elements/22/datafilter`

Request body:

```
<datafilter>
    <and>
        <condition key="[Lines]" operator="InList">
            <value>City Trousers</value>
            <value>Shirt Waist</value>
            <value>Jackets</value>
        </condition>
        <condition key="[State]" operator="NotEqual">
            <value>Colorado</value>
        </condition>
        <or>
            <condition key="[Sales revenue]" operator="Greater">
                <value>5000</value>
            </condition>
            <condition key="[Sales revenue]" operator="Less">
                <value>10</value>
            </condition>
        </or>
    </and>
</datafilter>
```

Response:

```
<success>
```

```
    <message>The resource of type "Report" with identifier "1" has been
successfully updated.</message>
    <id>1</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.19.4 Deleting a Data Filter from a Report Element

Usage

Deletes a data filter from a specific report element, referenced by its ID.

Request

```
DELETE /documents/<documentID>/reports/<reportID>/elements/<elementID>/datafilter
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/9512/reports/67/elements/22/datafilter
```

Response:

```
<success>
    <message>The resource of type "Report" with identifier "67" has been
successfully deleted.</message>
    <id>67</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the List of Report Elements [page 396]

8.4.20 Working with Sorts

Below are the tasks you can perform on the sorts of a report element:

- Getting, updating, or deleting the sorts of a report element
- Getting, updating, or deleting a specific sort of a report element

The sorts can be applied to the following report elements : Sections, Tables, Cells in Tables or Section Cell, Charts and Custom Elements.

8.4.20.1 Getting sorts in a report element

Usage

Gets the details of the sorts in a report element.

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/sorts
```

Response

Response type: `application/xml` or `application/json`

Example: 1. Radar chart (XML format)

```
[GET] /documents/8401/reports/4/elements/130/sorts
```

Response:

```
<sorts>
    <group>
        <name>Sort 1</name>
        <sort order="None">
            <id>R4.B130.S0</id>
            <assignedDataObject refId="DP0.DO5" kind="Expression"
dataType="String">=[Service]</assignedDataObject>
        </sort>
        <sort order="None">
            <id>R4.B130.S1</id>
            <assignedDataObject refId="DP0.DO25" kind="Expression"
dataType="Numeric">=[Future guests]</assignedDataObject>
        </sort>
    </group>
    <group>
        <name>Sort 2</name>
        <sort order="Ascending">
            <id>R4.B130.S2</id>
            <assignedDataObject refId="DP0.DO2" kind="Expression"
dataType="String">=[Resort]</assignedDataObject>
        </sort>
    </group>
</sorts>
```

Example: 2. Pie chart (XML format)

```
[GET] /documents/8579/reports/7/elements/289/sorts
```

Response:

```
<sorts>
    <sort order="Descending">
        <id>R7.B289.S0</id>
        <assignedDataObject refId="DP0.DO7" kind="Expression"
dataType="Numeric">=[Revenue]</assignedDataObject>
    </sort>
    <sort order="Ascending">
        <id>R7.B289.S1</id>
        <assignedDataObject refId="DP0.DO39" kind="Expression"
dataType="String">=[Country]</assignedDataObject>
    </sort>
    <sort order="None">
        <id>R7.B289.S2</id>
        <assignedDataObject refId="DP0.DO31" kind="Expression"
dataType="String">=[Year]</assignedDataObject>
    </sort>
</sorts>
```

Example: 3. Cross table (XML format)

```
[GET] /documents/8579/reports/7/elements/289/sorts
```

Response:

```
<sorts>
```

```
    <group>
        <name>Columns</name>
        <sort order="None">
            <id>R4.B12.S0</id>
            <assignedDataObject refId="DP0.DO39" kind="Expression"
dataType="String">=[Country]</assignedDataObject>
        </sort>
        <sort order="None">
            <id>R4.B12.S1</id>
            <assignedDataObject refId="DP0.DO31" kind="Expression"
dataType="String">=[Year]</assignedDataObject>
        </sort>
    </group>
    <group>
        <name>Rows</name>
        <sort order="None">
            <id>R4.B12.S2</id>
            <assignedDataObject refId="DP0.DO25" kind="Expression"
dataType="Numeric">=[Future guests]</assignedDataObject>
        </sort>
        <sort order="None">
            <id>R4.B12.S3</id>
            <assignedDataObject refId="DP0.DO5" kind="Expression"
dataType="String">=[Service]</assignedDataObject>
        </sort>
        <sort order="None">
            <id>R4.B12.S4</id>
            <assignedDataObject refId="DP0.DO4" kind="Expression"
dataType="String">=[Service Line]</assignedDataObject>
        </sort>
    </group>
</sorts>
```

Example: 4. Table cell (XML format)

```
[GET] /documents/8244/reports/1/elements/10/sorts
```

Response:

```
<sorts>
    <sort order="None">
        <id>R1.B12.S3</id>
        <assignedDataObject dataType="String">=NameOf([Year])</
assignedDataObject>
    </sort>
</sorts>
```

8.4.20.2 Getting the Details of a Sort

Usage

Gets the details of a sort in a report element.

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/sorts/ <sortID>
```

Response

Response type: `application/xml` or `application/json`

Parameters

- `documentId`: a valid identifier of a Webl document.
- `reportId`: a valid identifier of a report of the specified document.
- `elementId`: a valid identifier of a report element.
- `sortId`: a valid identifier of a sort.

Example: (XML format)

```
[GET] raylight/v1/documents/8486/reports/7/elements/289/sorts/R7.B289.S1
```

Response:

```
<sort order="Ascending">
    <id>R7.B289.S1</id>
    <assignedDataObject refId="DP0.DO39" kind="Expression"
dataType="String">=[Country]</assignedDataObject>
</sort>
```

8.4.20.3 Updating the sorts of a report element

Use this URL to update all the sorts of a report element in one shot. The position of each sort is significant, the sorts will be applied in the given order. Please notice that only the attributes "order" and the sorts position can be modified. You cannot move a sort from one group to another nor add a new sort. You can only modify the existing sorts.

Usage

Update all the sorts of a report element in one shot.

Request

```
PUT /documents/<documentID>/reports/<reportID>/elements/<elementID>/sorts
```

Response

Response type: `application/xml` or `application/json`

Parameters

- `documentId`: a valid identifier of a Web Intelligence document.
- `reportId`: a valid identifier of a report of the specified document.
- `elementId`: a valid identifier of a report element.

Example: 1. Cross table (XML format)

```
[PUT] /documents/9134/reports/4/elements/12/sorts
```

Request Body:

```
<sorts>
  <group>
    <sort order="Descending">
      <id>R4.B12.S1</id>
    </sort>
    <sort order="None">
      <id>R4.B12.S0</id>
    </sort>
  </group>
  <group>
    <sort order="Ascending">
      <id>R4.B12.S2</id>
    </sort>
    <sort order="Descending">
      <id>R4.B12.S4</id>
    </sort>
    <sort order="None">
      <id>R4.B12.S3</id>
    </sort>
  </group>
</sorts>
```

Response:

```
<success>
    <message>The resource of type "Report element" with identifier "12" has been
successfully updated.</message>
    <id>12</id>
</success>
```

Example: 2. Table cell (XML format)

```
[PUT] /documents/8263/reports/3/elements/10/sorts
```

Request Body:

```
<sorts>
    <sort order="Descending" />
</sorts>
```

Response:

```
<success>
    <message>The resource of type "Report element" with identifier "10" has been
successfully updated.</message>
    <id>10</id>
</success>
```

> **Note**
>
> Updating the sorts of a Section/Table Cell does not require you to specify the sort's ID, as there can only be one sort.

8.4.20.4 Updating a Sort in a Report Element

Use this URL to update a specific sort of a report element. Only the attribute `order` can be modified..

Usage

Update a specific sort of a report element.

Request

```
PUT /documents/<documentID>/reports/<reportID>/elements/<elementID>/sorts/ <sortID>
```

Response

Response type: `application/xml` or `application/json`

Parameters

- `documentId`: a valid identifier of a Web Intelligence document.
- `reportId`: a valid identifier of a report of the specified document.
- `elementId`: a valid identifier of a report element.
- `sortId`: a valid identifier of a report element.

Example: (XML format) (XTable)

```
[PUT] raylight/v1/documents/9274/reports/7/elements/270/sorts/R7.B270.S0
```

Request Body:

```
<sort order="Ascending" />
```

Response:

```
<success>
    <message>The resource of type "Sort" with identifier "R7.B270.S0" has been
successfully updated.</message>
    <id>R7.B270.S0</id>
</success>
```

8.4.20.5 Removing the Sorts of a Report Element

Use this URL to remove the sorts of a report element. This call will reset the sorts position and the attribute "order" to "None" for all the sorts.

Usage

Remove the sorts of a report element.

Request

```
DELETE /documents/<documentID>/reports/<reportID>/elements/<elementID>/sorts
```

Response

Response type: `application/xml` or `application/json`

Parameters

- `documentId`: a valid identifier of a Web Intelligence document.
- `reportId`: a valid identifier of a report of the specified document.
- `elementId`: a valid identifier of a report element.

Example: (XML format) (XTable)

`[GET] /documents/9134/reports/4/elements/12/sorts`

Request Body:

```
<sorts>
  <group>
    <sort order="Descending">
      <id>R4.B12.S1</id>
    </sort>
    <sort order="None">
      <id>R4.B12.S0</id>
    </sort>
  </group>
  <group>
    <sort order="Ascending">
      <id>R4.B12.S2</id>
    </sort>
    <sort order="Descending">
      <id>R4.B12.S4</id>
    </sort>
    <sort order="None">
      <id>R4.B12.S3</id>
    </sort>
  </group>
</sorts>
```

Response:

```
<success>
    <message>The resource of type "Report element" with identifier "12" has been
successfully updated.</message>
    <id>12</id>
</success>
```

8.4.20.6 Removing a specific sort of a report element

Use this URL to remove a specific sort of a report element. This call will reset the attribute "order" to "None" for the specified sort.

Usage

Remove the sorts of a specified report element.

Request

```
DELETE /documents/<documentID>/reports/<reportID>/elements/<elementID>/sorts/
<sortID>
```

Response

Response type: `application/xml` or `application/json`

Parameters

- `documentId`: a valid identifier of a Web Intelligence document.
- `reportId`: a valid identifier of a report of the specified document.
- `elementId`: a valid identifier of a report element.
- `sortId`: a valid identifier of a sort.

Example: (XML format)

```
[DELETE] raylight/v1/documents/9314/reports/2/elements/8/sorts/R2.B8.S1
```

Response:

```
<success>
    <message>The resource of type "Sort" with identifier "R2.B8.S1" has been
successfully removed.</message>
    <id>R2.B8.S1</id>
</success>
```

8.4.21 Managing Report Element Comments

Below are the tasks you can perform on report element comments:

- Get specific comments of a report element
- Get all the comments of a report element

Related Information

Getting a Specific Comment of a Report Element [page 495]
Getting all the Comments or Only the Visible Comments of a Report Element [page 493]

8.4.21.1 Getting all the Comments or Only the Visible Comments of a Report Element

Usage

This call returns all comments of a report element, or only the visible ones.

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/comments?
filter=visible
```

Note

Report element of type `PageZone` and `Section` cannot be commented. The call should throw an error, for example:

Sample Code

```
HTTP code: 400
<?xml version="1.0" encoding="UTF-8"?>
<error>
    <error_code>WSR 00101</error_code>
    <message>The resource of type "Report element" with identifier "2" does
not support inner resources of type "Comment" ("PageZone").</message>
</error>
```

Parameters

Required

- `<documentID>`: (type=integer) a valid identifier of a Web Intelligence document.
- `<reportID>`: (type=integer) a valid identifier of a report inside the Web Intelligence document.
- `<elementID>`: (type=integer) a valid identifier of an element inside a report.

Optional

- `<filter>`:(type=string) returns the comments you've as indicated on the call.
  Possible values are `all` (by default) or `visible`.
- `<datapath>`:(type=string) specifies the datapath that identifies the instance.
- `<reference>`:(type=string) specifies the reference of the instance on the element (when several instances exist).

> **Note**
>
> You can't use the `datapath` and `reference` parameters at the same time.

Response

Response type: `application/xml` or `application/json`

Example: XML

```
[GET] <url>/documents/22684/reports/19/elements/4/comments?
reference=J.4&filter=visible
```

If `?filter=visible`, the returned list in `<comments>` contains only one `<comment>`

Response:

```
<comments>
    <comment>
        <id>82</id>
        <created>2016-08-16T15:18:16.000Z</created>
        <updated>2016-08-16T15:18:16.000Z</updated>
        <createdBy>
            <name>fhsql_user2</name>
            <cuid>Aa6EkoIN4_xMhE2FhhaTEAA</cuid>
        </createdBy>
        <lastAuthor>
            <name>fhsql_user2</name>
            <cuid>Aa6EkoIN4_xMhE2FhhaTEAA</cuid>
        </lastAuthor>
        <value>123</value>
        <element>
            <id>7</id>
            <reference>J.7.0.E</reference>
        </element>
```

```
    </comment>
</comments>
```

8.4.21.2 Getting a Specific Comment of a Report Element

Usage

This call returns all comments of a report element of a Web Intelligence document, regardless of their visibility.

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/comments/
<commentID>
```

Parameters

Required

- `<documentID>`: (type=integer) a valid identifier of a Web Intelligence document.
- `<reportID>`: (type=integer) a valid identifier of a report inside the Web Intelligence document.
- `<elementID>`: (type=integer) a valid identifier of an element inside a report.
- `<commentID>`: (type=integer) a valid identifier of a comment.

Optional

- `<reference>`:(type=string) specify a reference.

Response

Response type: `application/xml` or `application/json`

Example: XML

```
[GET] <url>/documents/22684/reports/19/elements/4/comments/80?reference=J.4
```

Response:

```
<comment>
    <id>80</id>
```

```
    <created>2016-08-16T15:18:03.000Z</created>
    <updated>2016-08-16T15:18:03.000Z</updated>
    <createdBy>
        <name>raylight_user</name>
        <cuid>AZFmWPGBwwRBm3almUJGkfk</cuid>
    </createdBy>
    <lastAuthor>
        <name>raylight_user</name>
        <cuid>AZFmWPGBwwRBm3almUJGkfk</cuid>
    </lastAuthor>
    <value>Second</value>
    <element>
        <id>4</id>
        <reference>J.4</reference>
    </element>
</comment>
```

8.4.21.3 Adding a New Comment to a Report Element

Usage

Adds a new comment to a specific report element of a Web Intelligence document.

Request

```
POST /documents/<{documentId}>/reports/<{reportId}>/elements/<{elementId}>/comments
```

Parameters

Required

- `<documentId>`: (type=integer) a valid identifier of a Web Intelligence document.
- `<reportId>`: (type=integer) a valid identifier of a report inside the Web Intelligence document.
- `<elementId>`: (type=integer) a valid identifier of an element inside a report.

Optional

- `<datapath>`: (type=string) specifies the datapath of the instance on the element.
- `<reference>`: (type=string) specifies the reference of the instance on the element (when several instances exist).

> **Note**
>
> You can't use the `<datapath>` and `<reference>` parameters at the same time.

Response

Reponse type: `application/xml`

The response is a message stating the success or failure of the request.

Example: XML

`POST /documents/22684/reports/19/elements/5/comments`

Request:

```
<comment>
    <value>Great results in Q4!</value>
</comment>
```

Response:

```
<success>
    <message>The resource of type 'Comment' with identifier '5'  has been
successfully created.</message>
    <id>5</id>
<success>
```

8.4.21.4 Modfiying a Specific Message in a Comment Thread

Usage

Modifies a specific comment of a report element.

Request

`PUT documents/<{documentId}>/reports/<{reportId}>/elements/<{elementId}>/comments/<{commentId}>`

Parameters

Required

- `<documentId>`: (type=integer) a valid identifier of a Web Intelligence document.
- `<reportId>`: (type=integer) a valid identifier of a report inside the Web Intelligence document.

- `<elementId>`: (type=integer) a valid identifier of an element inside a report.
- `<commentId>`: (type=integer) a valid identifier of a comment.

Optional

- `<datapath>`: (type=string) specifies the datapath of the instance on the element.
- `<reference>`: (type=string) specifies the reference of the instance on the element (when several instances exist).

> **Note**
>
> You can't use the `<datapath>` and `<reference>` parameters at the same time.

Response

Reponse type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: XML

`[PUT] /documents/22684/reports/19/elements/4/comments/5`

Request:

```
<comment>
    <value>Update: Revenue covers country only</value>
</comment>
```

Response:

```
<success>
  <message>The resource of type 'Comment' with identifier '5' has been
successfully edited.</message>
  <id>5</id>
<success>
```

8.4.21.5 Deleting a Message in a Comment Thread

Usage

Deletes a specific message in the comment thread of a report element.

Request

```
DELETE /documents/<{docId}>/reports/<{repId}>/elements/<{elId}>/comments/
<{commentId}>
```

Parameters

- `<documentId>`: (type=integer) a valid identifier of a Web Intelligence document.
- `<reportId>`: (type=integer) a valid identifier of a report inside the Web Intelligence document.
- `<elementId>`: (type=integer) a valid identifier of an element inside a report.
- `<commentId>`: (type=integer) a valid identifier of a comment.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: XML

```
[DELETE] /documents/22684/reports/19/elements/4/comments/5
```

Reponse:

```
<success>
    <message>The resource of type "Comment" with identifier "5" has been
successfully deleted.</message>
<success>
```

8.5 Working With Input Controls

You can apply input controls to one report, a series of reports, or a whole document. You can also move an input control from a report to a document, and vice versa.

Related Information

Working with Document Input Controls [page 500]
Working with Report Input Controls [page 525]

8.5.1 Working with Document Input Controls

You can apply input controls to one report, a series of reports, or a whole document. You can also move an input control from a report to a document, and vice versa.

The following input controls are supported:

- TextField
- ComboBox
- RadioButtons
- CheckBox
- ListBox
- Calendar
- Spinner
- Slider
- TreeList
- Group of Input Controls

Getting the Input Controls of a Document [page 500]

Getting the Definition of an Input Control or a Group of Input Controls in a Document [page 504]

Updating the Definition of a Document Input Control [page 510]

Adding an Input Control to a Document [page 514]

Adding a New Group of Input Controls in a Document [page 515]

Getting the List of Values of a Document Input Control [page 516]

Getting the Selection of a Document Input Control [page 517]

Setting the Selection of a Document Input Control [page 519]

Deleting the Selection of a Document Input Control [page 520]

Deleting a Group of Input Controls from a Document [page 521]

Deleting a Document Input Control [page 522]

Reordering Input Controls in a Document [page 523]

Moving an Input Control from a Report to a Document [page 524]

8.5.1.1 Getting the Input Controls of a Document

Usage

Lists all the input controls of a document.

Request

```
GET /documents/<documentID>/inputcontrols
```

Optional parameter

`allInfo`: (type=boolean, default=false) if set to true, the details and the selection of each input control is included.

Response

Response type: `application/xml` or `application/json`

Response body: the list of input control identifiers and names.

Example: XML

```
GET /documents/6671/inputcontrols
```

Response:

```
<inputcontrols>
    <inputcontrol>
        <id>D.IF0</id>
        <name>Country</name>
    </inputcontrol>
</inputcontrols>
```

Example: JSON

```
GET /documents/5152/inputcontrols
```

Response:

```
{"inputcontrols":
    {"inputcontrol":
        [{"id":"D.IF0",
         "name":"my checkbox_1"}]
    }
}
```

Example: Example with details and selection (XML format):

```
GET /documents/7411/inputcontrols?allInfo=true
```

Response:

```
<inputcontrols>
    <inputcontrol>
        <id>D.IF0</id>
        <name>Country</name>
        <comboBox allowAllValuesSelection="true" useCustom="false"
operator="Equal"/>
        <assignedDataObject refId="DP0.DO39" kind="Expression"
dataType="String"/>
        <assignedReportElements>
            <assignedReportElement refId="D" reference="D.IF0.T0"/>
        </assignedReportElements>
        <selection>
            <value>US</value>
        </selection>
    </inputcontrol>
</inputcontrols>
```

Example: XML Format

```
GET /documents/8020/inputcontrols
```

Response

```
<inputcontrols>
    <inputcontrol>
        <id>R30.G0</id>
        <name>Location</name>
        <group>
            <id>R30.IF1</id>
            <id>R30.IF2</id>
        </group>
    </inputcontrol>
    <inputcontrol>
        <id>R30.IF3</id>
        <name>Name of manager</name>
    </inputcontrol>
    <inputcontrol>
        <id>R30.G1</id>
        <name>Weeks</name>
        <group>
            <id>R30.IF4</id>
            <id>R30.IF6</id>
            <id>R30.IF5</id>
        </group>
    </inputcontrol>
</inputcontrols>
```

The group above shows a group of two input controls, then a single input control, then a group of three input controls.

Example: With Details and Selection (XML format)

```
GET /documents/8020/inputcontrols?allInfo=true
```

Response

```
<inputcontrols>
    <inputcontrol>
        <id>R30.G0</id>
        <name>Location</name>
        <group hasFilterPath="true">
            <inputcontrol>
                <id>R30.IF1</id>
                <name>State</name>
                <listBox allowAllValuesSelection="true" useCustom="false"
numberOfLines="5" cardinality="Multiple" operator="InList"/>
                <assignedDataObject refId="DP0.DOda" kind="Expression"
dataType="String"/>
                <assignedReportElements>
                    <assignedReportElement refId="29" reference="R30.IF1.T0"/>
                    <assignedReportElement refId="22" reference="R30.IF1.T1"/>
                </assignedReportElements>
                <groupingInfo refId="R30.G0" inFilterPath="true"
eligibility="Full"/>
                <selection>
                    <value>California</value>
                    <value>DC</value>
                    <value>Illinois</value>
                </selection>
            </inputcontrol>
            <inputcontrol>
                <id>R30.IF2</id>
                <name>City</name>
                <listBox allowAllValuesSelection="true" useCustom="false"
numberOfLines="5" cardinality="Multiple" operator="InList"/>
                <assignedDataObject refId="DP0.DOa6" kind="Expression"
dataType="String"/>
                <assignedReportElements>
                    <assignedReportElement refId="29" reference="R30.IF2.T0"/>
                    <assignedReportElement refId="22" reference="R30.IF2.T1"/>
                </assignedReportElements>
                <groupingInfo refId="R30.G0" inFilterPath="true"
eligibility="Full"/>
                <selection>
                    <value>Chicago</value>
                    <value>San Francisco</value>
                    <value>Washington</value>
                </selection>
            </inputcontrol>
        </group>
    </inputcontrol>
    <inputcontrol>
        <id>R30.IF3</id>
        <name>Name of manager</name>
        <listBox allowAllValuesSelection="true" useCustom="false"
numberOfLines="5" cardinality="Multiple" operator="InList"/>
        <assignedDataObject refId="DP0.DO3c" kind="Expression"
dataType="String"/>
        <assignedReportElements>
            <assignedReportElement refId="29" reference="R30.IF3.T0"/>
        </assignedReportElements>
        <groupingInfo eligibility="Full"/>
        <selection>
            <value>Barrett</value>
            <value>Steve</value>
        </selection>
```

```
    </inputcontrol>
    ...
</inputcontrols>
```

With this option, the input controls inside a group are fully expanded, whereas without the option, only the IDs are returned.

8.5.1.2 Getting the Definition of an Input Control or a Group of Input Controls in a Document

Usage

Retrieve the input control or group input control definition.

Request

```
GET /documents/{documentId}/inputcontrols/{inputcontrolId}
```

Parameters

`inputcontrolId`: a valid input control identifier

Response Type

Response type: `application/xml` or `application/json`

Example: XML Format

```
GET /documents/8020/inputcontrols/R2.IF1
```

Response

```
<inputcontrol>
    <id>R2.IF1</id>
    <name>Quarter</name>
    <listBox allowAllValuesSelection="true" useCustom="false" numberOfLines="5"
cardinality="Multiple" operator="InList"/>
```

```
    <assignedDataObject refId="DP0.DOba" kind="Expression" dataType="String"/>
    <assignedReportElements>
        <assignedReportElement refId="17" reference="R2.IF1.T0"/>
    </assignedReportElements>
    <groupingInfo refId="R2.G0" inFilterPath="true" eligibility="Full"/>
</inputcontrol>
```

The `<groupingInfo>` gives details about the grouping state with the following attributes:

- `refId` and `inFilterPath` are only present of the input control in included in a group and give the group ID and if the input control is used in a filter path
- `eligibility` is always present and can take the following values:
  - `Full`: The input control can be added into a group input control
  - `Restricted`: The input control can be added into a group input control but its filtering will be limited
  - `None`: The input control cannot be added to a group

Example: A group of input controls (XML format)

```
GET /documents/8020/inputcontrols/D.GO
```

Response

```
<inputcontrol>
    <id>D.G0</id>
    <name>Months</name>
    <group hasFilterPath="true">
        <id>D.IF0</id>
        <id>D.IF1</id>
    </group>
</inputcontrol>
```

Example: Combo Box (XML)

```
GET /documents/6671/inputcontrols/D.IF0
```

Response

```
<inputcontrol>
    <id>D.IF0</id>
    <name>Country</name>
    <comboBox allowAllValuesSelection="true" useCustom="false"
operator="Equal"/>
    <assignedDataObject refId="DP0.DO39" kind="Expression"/>
    <assignedReportElements>
        <assignedReportElement refId="D" reference="D.IF0.T0"/>
    </assignedReportElements>
</inputcontrol>
```

Example: CheckBox (JSON)

```
GET /documents/5152/inputcontrols/D.IF0
```

Response:

```
{"inputcontrol":
    {"id":"D.IF0",
     "name":"my checkbox_1",
     "checkBox":
        {"@numberOfLines":"5",
         "@useCustom":"false",
         "@allowAllValuesSelection":"true",
         "@operator":"InList"},
     "assignedDataObject":
        {"@kind":"Expression",
         "@refId":"DP0.DO1fc"},
     "assignedReportElements":
        {"assignedReportElement":
            [{"@reference":"D.IF0.T0",
             "@refId":"D"}]
        }
    }
}
```

Example: Example with datatype details and selection (XML format):

Since 4.2 SP4, the data type of the assigned data object is also returned in the output (but skipped on input for input control creation or update.

```
GET /documents/7258/inputcontrols/D.IF0
```

Response:

```
<inputcontrol>
    <id>D.IF0</id>
    <name>Year</name>
    <listBox allowAllValuesSelection="true" useCustom="false" numberOfLines="5"
cardinality="Multiple" operator="InList"/>
    <assignedDataObject refId="DP0.DObc" kind="Expression" dataType="String"/>
    <assignedReportElements>
        <assignedReportElement refId="D" reference="D.IF0.T0"/>
    </assignedReportElements>
    <groupId>D.G0</groupId>
</inputcontrol>
```

Example: Example with grouping information (XML format):

Since 4.2 SP4, a new element related to grouping information has been added.

```
GET /documents/7258/inputcontrols/D.IF0
```

Response:

```
<inputcontrol>
```

```
    <id>D.IF0</id>
    <name>Year</name>
    <listBox allowAllValuesSelection="true" useCustom="false" numberOfLines="5"
cardinality="Multiple" operator="InList"/>
    <assignedDataObject refId="DP0.DObc" kind="Expression" dataType="String"/>
    <assignedReportElements>
        <assignedReportElement refId="D" reference="D.IF0.T0"/>
    </assignedReportElements>
    <groupId>D.G0</groupId>
</inputcontrol>
```

Example: Example with grouping information including the 'eligibility' attribute (XML format):

```
[GET] documents/7258/inputcontrols/D.IF0
```

Response:

```
<inputcontrol>
    <id>D.IF0</id>
    ...
    <groupingInfo refId="D.G0" inFilterPath="true" eligibility="Full"/>
</inputcontrol>
```

Values for the `eligibility` attribute are:

- `None`
- `Restricted`
- `Full`

8.5.1.2.1 Getting the Definition of a Document Input Control

Usage

Returns the details of an input control of a document.

Request

```
GET /documents/<documentID>/inputcontrols/<inputControlID>
```

Response

Response type: `application/xml` or `application/json`

> **Note**
>
> Since 4.2 SP4, a new '`allowNullValueSelection`' boolean attribute has been added to the widget attributes list for '`ComboBox`', '`ListBox`', '`CheckBox`', and '`RadioButtons`'.

Example: Combo Box (XML)

```
GET /documents/6671/inputcontrols/D.IF0
```

Response

```
<inputcontrol>
    <id>D.IF0</id>
    <name>Country</name>
    <comboBox allowAllValuesSelection="true" useCustom="false"
operator="Equal"/>
    <assignedDataObject refId="DP0.DO39" kind="Expression"/>
    <assignedReportElements>
        <assignedReportElement refId="D" reference="D.IF0.T0"/>
    </assignedReportElements>
</inputcontrol>
```

Example: CheckBox (JSON)

```
GET /documents/5152/inputcontrols/D.IF0
```

Response:

```
{"inputcontrol":
    {"id":"D.IF0",
     "name":"my checkbox_1",
     "checkBox":
        {"@numberOfLines":"5",
         "@useCustom":"false",
         "@allowAllValuesSelection":"true",
         "@operator":"InList"},
     "assignedDataObject":
        {"@kind":"Expression",
         "@refId":"DP0.DO1fc"},
     "assignedReportElements":
        {"assignedReportElement":
            [{"@reference":"D.IF0.T0",
             "@refId":"D"}]
        }
    }
}
```

Example: Example with datatype details and selection (XML format):

Since 4.2 SP4, the data type of the assigned data object is also returned in the output (but skipped on input for input control creation or update.

```
[GET] documents/7258/inputcontrols/D.IF0
```

Response:

```
<inputcontrol>
    <id>D.IF0</id>
    <name>Year</name>
    <listBox allowAllValuesSelection="true" useCustom="false" numberOfLines="5"
cardinality="Multiple" operator="InList"/>
    <assignedDataObject refId="DP0.DObc" kind="Expression" dataType="String"/>
    <assignedReportElements>
        <assignedReportElement refId="D" reference="D.IF0.T0"/>
    </assignedReportElements>
    <groupId>D.G0</groupId>
</inputcontrol>
```

Example: Example with grouping information (XML format):

Since 4.2 SP4, a new element related to grouping information has been added.

```
[GET] documents/7258/inputcontrols/D.IF0
```

Response:

```
<inputcontrol>
    <id>D.IF0</id>
    <name>Year</name>
    <listBox allowAllValuesSelection="true" useCustom="false" numberOfLines="5"
cardinality="Multiple" operator="InList"/>
    <assignedDataObject refId="DP0.DObc" kind="Expression" dataType="String"/>
    <assignedReportElements>
        <assignedReportElement refId="D" reference="D.IF0.T0"/>
    </assignedReportElements>
    <groupId>D.G0</groupId>
</inputcontrol>
```

Example: Example with grouping information including the 'eligibility' attribute (XML format):

```
[GET] documents/7258/inputcontrols/D.IF0
```

Response:

```
<inputcontrol>
    <id>D.IF0</id>
    ...
    <groupingInfo refId="D.G0" inFilterPath="true" eligibility="Full"/>
</inputcontrol>
```

Values for the `eligibility` attribute are:

- `None`
- `Restricted`
- `Full`

Related Information

Getting the List of Documents [page 226]
Getting the Input Controls of a Document [page 500]

8.5.1.3 Updating the Definition of a Document Input Control

Usage

Updates the details of an input control of a document, such as a name or description.

Request

```
PUT /documents/<documentID>/inputcontrols/<inputControlID>
```

Request type: `application/xml` or `application/json`

> **Note**
>
> Since 4.2 SP04, a new '`allowNullValueSelection`' boolean attribute has been added to the widget attributes list for 'ComboBox', 'ListBox', 'CheckBox', and 'RadioButtons'.

The update of an input control must adhere to the following rules:

- You cannot change the widget of an input control.
- The input control must have at least one assigned report element.
- You can change the assigned data object, but it must be compatible with the widget.
- You can change the operator, but it must be compatible with the widget.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/6671/inputcontrols/D.IF0
```

Request body:

```
<inputcontrol>
    <name>V_update</name>
    <comboBox allowAllValuesSelection="true" useCustom="false"
operator="GreaterOrEqual"/>
    <assignedDataObject refId="DP0.DO2"/>
</inputcontrol>
```

> **Note**
>
> Since 4.2 SP4, a new '`allowNullValueSelection`' boolean attribute has been added to the widget attributes list for '`ComboBox`', '`ListBox`', '`CheckBox`', and '`RadioButtons`'.

Response:

```
<success>
    <message>The resource of type "Input Control" with identifier "D.IF0" has
been successfully updated.</message>
    <id>D.IF0</id>
</success>
```

8.5.1.3.1 Updating the Definition of an Input Control or Group of Input Controls in a Document

Usage

Update the definition of an existing input control or group of input controls.

For a regular input control attached to a group, in addition to the already known operations, it is also possible to remove it from the filter path.

> **Note**
>
> Since 4.2 SP4

Request

```
[PUT] <url>/documents/{documentId}/inputcontrols/{inputcontrolId}
```

Parameters

`inputcontrolId`: a valid input control identifier

Content Type

Content type: `application/xml` or `application/json`

Response Type

Response type: `application/xml` or `application/json`

Example: Remove from Filter Path an Input Control Attached to a Group (XML format)

```
[PUT] <url>/documents/8020/inputcontrols/R30.IF1
```

Request

```
<inputcontrol>
        <groupingInfo inFilterPath="false"/>
</inputcontrol>
```

Response

```
<success>
    <message>The resource of type "Input Control" with identifier "R30.IF1" has
been successfully updated.</message>
    <id>R30.IF1</id>
</success>
```

Example: Change the Name of a Group of Input Controls (XML format)

Groups of input controls may also be updated:

- To change their name
- To change the input controls list
- To reset the whole filter path.

```
[PUT] <url>/documents/8020/inputcontrols/R1.G0
```

> **Note**
>
> The operations in the examples below may be combined in a single request.

Request

```
<inputcontrol>
        <name>New group name</name>
</inputcontrol>
```

Response

```
<success>
    <message>The resource of type "Input Control" with identifier "R1.G0" has
been successfully updated.</message>
    <id>R1.G0</id>
</success>
```

Example: Change the Regular Input Controls Attached to this Group (XML format)

```
[PUT] <url>/documents/8020/inputcontrols/R1.G0
```

Request

```
<inputcontrol>
    <group>
        <id>R1.IF2</id>
        <id>R1.IF3</id>
    </group>
</inputcontrol>
```

Response

```
<success>
    <message>The resource of type "Input Control" with identifier "R1.G0" has
been successfully updated.</message>
    <id>R1.G0</id>
</success>
```

Example: Reset the Filter Path (XML format)

```
[PUT] <url>/documents/8020/inputcontrols/R30.G3
```

Request

```
<inputcontrol>
    <group hasFilterPath='false'/>
</inputcontrol>
```

Response

```
<success>
    <message>The resource of type "Input Control" with identifier "R30.G3" has
been successfully updated.</message>
    <id>R30.G3</id>
</success>
```

8.5.1.4 Adding an Input Control to a Document

Usage

Adds a new input control to a document.

Request

`POST /documents/<documentID>/inputcontrols`

The request body must adhere to the following rules:

- The widget must be compatible with the assigned data object, for example: slide on a measure, radio buttons on a dimension.
- The widget must be compatible with the operator.

> **Note**
>
> Since 4.2 SP4, a new 'allowNullValueSelection' boolean attribute has been added to the widget attributes list for 'ComboBox', 'ListBox', 'CheckBox', and 'RadioButtons'.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: XML

`POST /documents/6671/inputcontrols`

Request body:

```
<inputcontrol>
    <name>my checkbox_1</name>
```

```
    <checkBox allowAllValuesSelection="true" useCustom="false"
operator="InList"/>
    <assignedDataObject refId="DP0.DO39"/>
</inputcontrol>
```

Response:

```
<success>
    <message>The resource of type "Input Control" with identifier "D.IF1" has
been successfully created.</message>
    <id>D.IF1</id>
</success>
```

Example: JSON

```
POST /documents/5152/inputcontrols
```

Request body:

```
{"inputcontrol":{
    "name":"my checkbox_1",
    "checkBox":
     {"@allowAllValuesSelection":"true",
      "@useCustom":"false",
      "@operator":"InList"},
    "assignedDataObject":{"@refId":"DP0.DO1fc"}
    }
}
```

Response:

```
{"success":
    {"message":"The resource of type \"Input Control\" with identifier \"D.IF0\"
has been successfully created.",
     "id":"D.IF0"}
}
```

8.5.1.5 Adding a New Group of Input Controls in a Document

Usage

Move eligible input controls into a group.

> **Note**
>
> Since 4.2 SP4

Request

```
[POST] <url>/documents/{documentId}/inputcontrols
```

Content Type

Content type: `application/xml` or `application/json`

Response Type

Response type: `application/xml` or `application/json`

Example: XML Format

```
POST /documents/8020/inputcontrols
```

Request

```
<inputcontrol>
    <name>New group</name>
    <group>
        <id>R30.IF1</id>
        <id>R30.IF4</id>
        <id>R30.IF3</id>
    </group>
</inputcontrol>
```

Response

```
<success>
    <message>The resource of type "Input Control" with identifier "R30.G2" has
been successfully created.</message>
    <id>R30.G2</id>
</success>
```

8.5.1.6 Getting the List of Values of a Document Input Control

New in 4.2 SP04, input controls URLs have now their own " /lov " calls. They are designed to return the list of values of an input control and set the input control selection.

Request

```
GET /documents/<documentID>/inputcontrols/<inputcontrolID>/lov
```

Response

Response type: `application/xml` or `application/json`

Response body: the list of values of the input control.

Example: Get the List of Values of an input control (XML format)

```
GET /documents/7679/inputcontrols/D.IF0/lov
```

Response:

```
<lov hierarchical="false" partial="false" refreshable="false" searchable="true"
mandatorySearch="false" searchScopes="Values" searchTargets="Server">
       <values>
             <value>France</value>
             <value>US</value>
       </values>
       <columns mappingId="0">
             <column id="0" type="String">Country</column>
       </columns>
</lov>
```

8.5.1.7 Getting the Selection of a Document Input Control

Usage

Gets the values selected for an input control of a document.

Request

```
GET /documents/<documentID>/inputcontrols/<inputControlID>/selection
```

Response

Response type: `application/xml` or `application/json`

The response is the list of values selected for the input control.

Example: XML

```
GET /documents/6671/inputcontrols/D.IF0/selection
```

Response:

```
<selection>
    <value>US</value>
</selection>
```

Response if all values are selected:

```
<selection all="true"/>
```

Response if no value or null value is selected:

```
<selection/>
```

Example: JSON

```
GET /documents/5152/inputcontrols/D.IF0/selection
```

Response:

```
{"selection":
    {"value":["Bermudas","Boatwear"]}
    }
```

Response if all values are selected:

```
{"selection":
    {"@all":"true",
     "value":[]}
    }
```

Response if no value or null value is selected:

```
{"selection":
    {"value":[]}
    }
```

8.5.1.8 Setting the Selection of a Document Input Control

Usage

Sets the values selected for an input control of a document.

Request

`PUT /documents/<documentID>/inputcontrols/<inputControlID>/selection`

Request body: the values can be obtained by getting the list of values from the assigned data object using the following call:

```
GET/documents/<documentID>/dataobjects/<dataObjectID>/lov
```

The `<selection>` can contain the `all="true"` attribute to specify that all values are selected.

The `<selection/>` tag specifies that no value or null values are selected.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: XML

`PUT /documents/6671/inputcontrols/D.IF0/selection`

Request body:

```
<selection>
    <value>France</value>
    <value>Germany</value>
</selection>
```

Request body with all values:

```
<selection all="true"/>
```

Request body with no value or null value:

```
<selection/>
```

Response:

```
<success>
    <message>The resource of type "Input Control Selection" with identifier
"D.IF0" has been successfully updated.</message>
    <id>D.IF0</id>
</success>
```

Example: JSON

```
PUT /documents/5152/inputcontrols/D.IF0/selection
```

Request body:

```
{"selection":
    {"value":["Bermudas","Boatwear"]}
    }
```

Request body with all values:

```
{"selection":
    {"@all":"true",
     "value":[]}
    }
```

Request body with no value or null value:

```
{"selection":
    {"value":[]}
    }
```

Response:

```
{"success":
    {"message":"The resource of type \"Input Control Selection\" with identifier
\"D.IF0\" has been successfully updated.",
     "id":"D.IF0"}
}
```

Related Information

Getting the List of Values of a Data Object [page 596]

8.5.1.9 Deleting the Selection of a Document Input Control

Usage

Removes the values selected from an input control of a document.

Request

```
DELETE /documents/<documentID>/inputcontrols/<inputControlID>/selection
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/6671/inputcontrols/D.IF0/selection
```

Response:

```
<success>
    <message>The resource of type "Input Control Selection" has been
successfully removed.</message>
</success>
```

8.5.1.10 Deleting a Group of Input Controls from a Document

Usage

This call removes a group of input controls (without deleting the input controls it contains).

> **Note**
>
> Since 4.2 SP4

Request

```
[DELETE] <url>/documents/{documentId}/inputcontrols/{inputcontrolId}
```

Parameters

`inputcontrolId`: a valid input control identifier

Response Type

Response type: `application/xml` or `application/json`

Example: XML format

```
[DELETE] <url>/documents/8020/D.G0
```

Response

```
<success>
    <message>The resource of type "Input Control" with identifier "D.G0" has
been successfully removed.</message>
    <id>D.G0</id>
</success>
```

8.5.1.11 Deleting a Document Input Control

Usage

Removes an input control from a document.

Request

```
DELETE /documents/<documentID>/inputcontrols/<inputControlID>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/6671/inputcontrols/D.IF0
```

Response:

```
<success>
    <message>The resource of type "Input Control" with identifier "D.IF0" has
been successfully removed.</message>
    <id>D.IF0</id>
</success>
```

8.5.1.12 Reordering Input Controls in a Document

Usage

Changes the internal input control indexes to reorder them.

Request

```
PUT /documents/<documentId>/inputcontrols
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example (XML)

```
PUT /documents/18729/inputcontrols
```

Request:

```
<inputcontrols>
    <inputcontrol><id>D.IF4</id></inputcontrol>
    <inputcontrol><id>D.IF3</id></inputcontrol>
</inputcontrols>
```

Response:

```
<success>
    <message>The resource of type "Document" with identifier "18729" has been
successfully updated.</message>
    <id>18729</id>
</success>
```

8.5.1.13 Moving an Input Control from a Report to a Document

Usage

Moves an input control from a report to a document. The input control obtains a new identifier after the move, but keeps the same data objects.

Request

`PUT /documents/<documentID>/inputcontrols/<inputControlID>?fromId=<fromID>`

Where:

- `<fromID>` is the input control ID in the report

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`PUT documents/6660/inputcontrols?fromId=R1.IF3`

Response:

```
<success>
    <message>The resource of type "Input Control" with identifier "R1.IF3" has
been successfully moved.</message>
    <id>D.IF0</id>
</success>
```

The `<id>` is the new identifier of the input control in the document.

Another call to retrieve the definition of the input control shows a fake report element of identifier **D** associated with the control in the document.

```
GET documents/6660/inputcontrols/D.IF0
```

Response:

```
<inputcontrol>
    <id>D.IF0</id>
    <name>Revenue</name>
    <slider minValue="30000.0" maxValue="600000.0" increment="1000.0"
cardinality="Interval" operator="Between">
        <default>
            <value>30000</value>
            <value>600000</value>
        </default>
    </slider>
    <assignedDataObject refId="DP0.DO7" kind="Expression"/>
    <assignedReportElements>
        <assignedReportElement refId="D" reference="D.IF0.T0"/>
    </assignedReportElements>
</inputcontrol>
```

8.5.2 Working with Report Input Controls

You can apply input controls to one report, a series of reports, or a whole document. You can also move an input control from a report to a document, and vice versa.

> **Note**
>
> Cascading input controls are now supported from 4.2 SP04.

The following input controls are supported:

- TextField
- ComboBox
- RadioButtons
- CheckBox
- ListBox
- Calendar
- Spinner
- Slider
- TreeList
- Group of Input Controls

Input Controls and Lists of Values [page 526]

Handling element linking (Block input controls) [page 527]

Getting the Input Controls of a Report [page 529]

Getting the Definition of an Input Control [page 533]

Updating the Definition of an Input Control [page 539]

Adding an Input Control [page 543]

Deleting an Input Control [page 545]
Getting the Selection of an Input Control [page 546]
Setting the Selection of an Input Control [page 547]
Deleting the Selection of an Input Control [page 548]
Reordering Input Controls in a Report [page 549]
Moving an Input Control from a Document to a Report [page 550]
Adding a New Group of Input Controls to a Report [page 552]
Deleting a Group of Input Controls from a Report [page 553]

8.5.2.1 Input Controls and Lists of Values

Regular input controls URLs have now their own " /lov " (list of values) calls. They were designed to handle groups filter paths, but are usable with all kinds of input controls, whether in a group or not.

Example: Get the list of values of an input control (XML format)

```
[GET] <url>/documents/8020/reports/48/inputcontrols/R48.IF2/lov
```

Response:

```
<lov hierarchical="false" partial="false" refreshable="false" searchable="true"
mandatorySearch="false">
    <values>
        <value>Accessories</value>
        <value>City Skirts</value>
        <value>City Trousers</value>
        <value>Dresses</value>
        <value>Jackets</value>
        <value>Leather</value>
        <value>Outerwear</value>
        <value>Overcoats</value>
        <value>Shirt Waist</value>
        <value>Sweaters</value>
        <value>Sweat-T-Shirts</value>
        <value>Trousers</value>
    </values>
    <columns mappingId="0">
        <column id="0" type="String">Lines</column>
    </columns>
</lov>
```

Example: Get the list of values of an input control, in descending order, and matching a pattern (XML format)

```
[PUT] <url>/documents/8020/reports/1/inputcontrols/R1.IF3/lov
```

Request body:

```
<lov>
  <query>
    <sort order='Descending' />
    <search>*a*</search>
  </query>
</lov>
```

Response:

```
<lov hierarchical="false" partial="false" refreshable="false" searchable="true"
mandatorySearch="false">
    <values>
        <value>Richards</value>
        <value>Mark</value>
        <value>Leonard</value>
        <value>Larry</value>
        <value>Barrett</value>
        <value>Anderson</value>
    </values>
    <columns mappingId="0">
        <column id="0" type="String">Name of manager</column>
    </columns>
</lov>
```

8.5.2.2 Handling element linking (Block input controls)

Usage

Lists all the input controls of a report. Element links are handled in the same way as other input controls, except the widget type which is "block" instead of more common "listBox", "slider", etc. The links are always report related, i.e. no document block input control is possible. Element links use a table, a chart, or a custom element, as source for filtering. Only expressions (or variables or links) of the source element, qualified as Dimension are used. Measures and Details are not used. An element link uses either one or all dimensions of a source report elements: using two dimensions out of three, for example, is not possible.

URLs are the same as for other report input controls.

Request

```
GET /documents/<documentID>/reports/<reportID>/inputcontrols
```

Example: Get the definition of an element link (single dimension)

```
[GET] /documents/7919/reports/22/inputcontrols/R22.IF0
```

Response:

```
<inputcontrol>
    <id>R22.IF0</id>
    <name>State</name>
    <block elementId="29" operator="Equal"/>
    <assignedDataObject refId="DP0.DOda" kind="Expression" dataType="String"/>
    <assignedReportElements>
        <assignedReportElement refId="46" reference="R22.IF0.T0"/>
    </assignedReportElements>
    <groupingInfo eligibility="None"/>
</inputcontrol>
```

Values for the `eligibility` attribute are:

- `None`
- `Restricted`
- `Full`

Example: Get the definition of an element link (all dimensions)

```
GET /documents/7411/reports/1/inputcontrols?allInfo=true
```

Response:

```
<inputcontrol>
    <id>R49.IF0</id>
    <name>Bloc 2</name>
    <block allowNullValueSelection="true" elementId="46" operator="Equal"/>
    <assignedDataObjects>
        <assignedDataObject refId="DP0.DOda" kind="Expression"
dataType="String"/>
        <assignedDataObject refId="DP0.DOb9" kind="Expression"
dataType="Numeric"/>
        <assignedDataObject refId="DP0.DOba" kind="Expression"
dataType="String"/>
        <assignedDataObject refId="DP0.DO178" kind="Expression"
dataType="String"/>
        <assignedDataObject refId="DP0.DO104" kind="Expression"
dataType="Numeric"/>
        <assignedDataObject refId="DP0.DObc" kind="Expression"
dataType="String"/>
        <assignedDataObject refId="DP0.DObb" kind="Expression"
dataType="String"/>
        <assignedDataObject refId="L1" kind="Variable" dataType="String"/>
    </assignedDataObjects>
    <assignedReportElements>
        <assignedReportElement refId="29" reference="R49.IF0.T0"/>
    </assignedReportElements>
    <groupingInfo eligibility="None"/>
</inputcontrol>
```

> **Note**
>
> For "all dimensions" element links, The single `<assignedDataObject>` element is superseded by an `<assignedDataObjects>` list.

8.5.2.3 Getting the Input Controls of a Report

Usage

Lists all the input controls of a report.

Request

```
GET /documents/<documentID>/reports/<reportID>/inputcontrols
```

Optional parameter(s)

`allInfo`: (type=boolean, default=false) if set to true, the details and the selection of each input control is included. Since 4.2 SP04.

Response

Response type: `application/xml` or `application/json`

Response body: the list of input control identifiers and names.

Example

```
GET /documents/6452/reports/1/inputcontrols
```

Response:

```
<inputcontrols>
    <inputcontrol>
        <id>R1.IF0</id>
        <name>Country</name>
    </inputcontrol>
    <inputcontrol>
        <id>R1.IF1</id>
```

```
        <name>Service Line</name>
    </inputcontrol>
    <inputcontrol>
        <id>R1.IF2</id>
        <name>Year</name>
    </inputcontrol>
    <inputcontrol>
        <id>R1.IF3</id>
        <name>Revenue</name>
    </inputcontrol>
    <inputcontrol>
        <id>R1.IF4</id>
        <name>var1_serviceline</name>
    </inputcontrol>
    <inputcontrol>
        <id>R1.IF5</id>
        <name>var2_revenue</name>
    </inputcontrol>
</inputcontrols>
```

Example: Example with details and selection (XML format): (since 4.2 SP4)

```
GET /documents/7411/reports/1/inputcontrols?allInfo=true
```

Response:

```
<inputcontrols>
    <inputcontrol>
        <id>R1.IF0</id>
        <name>Country</name>
        <checkBox allowAllValuesSelection="true" useCustom="false"
numberOfLines="5" operator="InList">
            <default>
                <value>France</value>
                <value>US</value>
            </default>
        </checkBox>
        <assignedDataObject refId="DP0.DO6" kind="Expression"
dataType="String"/>
        <assignedReportElements>
            <assignedReportElement refId="15" reference="R1.IF0.T0"/>
        </assignedReportElements>
        <selection>
            <value>France</value>
            <value>US</value>
        </selection>
    </inputcontrol>
    ...
    <inputcontrol>
        <id>R1.IF5</id>
        <name>var2_revenue</name>
        <spinner minValue="10.0" maxValue="15.0" increment="1.0"
operator="Equal">
            <default>
                <value>11</value>
            </default>
        </spinner>
        <assignedDataObject refId="L2" kind="Variable" dataType="Numeric"/>
        <assignedReportElements>
            <assignedReportElement refId="20" reference="R1.IF5.T0"/>
        </assignedReportElements>
        <selection>
            <value>13</value>
```

```
        </selection>
    </inputcontrol>
</inputcontrols>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.5.2.3.1 Getting the list of Input Controls for a Report

Usage

Returns the list of input controls of a report.

> **Note**
>
> Since 4.2 SP4

Request

```
[GET] <url>/documents/{documentId}/reports/{reportId}/inputcontrols
```

Optional parameter

`allInfo`: (type=boolean, default=false) if set to true, the details and the selection of each inputcontr ol is included.

Response

Response type: `application/xml` or `application/json`

Example: XML Format

```
GET /documents/8020/reports/30/inputcontrols
```

Response

```
<inputcontrols>
    <inputcontrol>
        <id>R30.G0</id>
        <name>Location</name>
        <group>
            <id>R30.IF1</id>
            <id>R30.IF2</id>
        </group>
    </inputcontrol>
    <inputcontrol>
        <id>R30.IF3</id>
        <name>Name of manager</name>
    </inputcontrol>
    <inputcontrol>
        <id>R30.G1</id>
        <name>Weeks</name>
        <group>
            <id>R30.IF4</id>
            <id>R30.IF6</id>
            <id>R30.IF5</id>
        </group>
    </inputcontrol>
</inputcontrols>
```

The group above shows a group of two input controls, then a regular input control, then a group of three input controls.

Example: With Details and Selection (XML format)

```
[GET] <url>/documents/8020/reports/30/inputcontrols?allInfo=true
```

Response

```
<inputcontrols>
    <inputcontrol>
        <id>R30.G0</id>
        <name>Location</name>
        <group hasFilterPath="true">
            <inputcontrol>
                <id>R30.IF1</id>
                <name>State</name>
                <listBox allowAllValuesSelection="true" useCustom="false"
numberOfLines="5" cardinality="Multiple" operator="InList"/>
                <assignedDataObject refId="DP0.DOda" kind="Expression"
dataType="String"/>
                <assignedReportElements>
                    <assignedReportElement refId="29" reference="R30.IF1.T0"/>
                    <assignedReportElement refId="22" reference="R30.IF1.T1"/>
                </assignedReportElements>
                <groupingInfo refId="R30.G0" inFilterPath="true"
eligibility="Full"/>
                <selection>
                    <value>California</value>
                    <value>DC</value>
```

```
                    <value>Illinois</value>
                </selection>
            </inputcontrol>
            <inputcontrol>
                <id>R30.IF2</id>
                <name>City</name>
                <listBox allowAllValuesSelection="true" useCustom="false"
numberOfLines="5" cardinality="Multiple" operator="InList"/>
                <assignedDataObject refId="DP0.DOa6" kind="Expression"
dataType="String"/>
                <assignedReportElements>
                    <assignedReportElement refId="29" reference="R30.IF2.T0"/>
                    <assignedReportElement refId="22" reference="R30.IF2.T1"/>
                </assignedReportElements>
                <groupingInfo refId="R30.G0" inFilterPath="true"
eligibility="Full"/>
                <selection>
                    <value>Chicago</value>
                    <value>San Francisco</value>
                    <value>Washington</value>
                </selection>
            </inputcontrol>
        </group>
    </inputcontrol>
    <inputcontrol>
        <id>R30.IF3</id>
        <name>Name of manager</name>
        <listBox allowAllValuesSelection="true" useCustom="false"
numberOfLines="5" cardinality="Multiple" operator="InList"/>
        <assignedDataObject refId="DP0.DO3c" kind="Expression"
dataType="String"/>
        <assignedReportElements>
            <assignedReportElement refId="29" reference="R30.IF3.T0"/>
        </assignedReportElements>
        <groupingInfo eligibility="Full"/>
        <selection>
            <value>Barrett</value>
            <value>Steve</value>
        </selection>
    </inputcontrol>
    ...
</inputcontrols>
```

With this option, the input controls inside a group are fully expanded, whereas without the option, only the ids are returned.

8.5.2.4 Getting the Definition of an Input Control

Usage

Returns the details of an input control of a report.

Request

```
GET /documents/<documentID>/reports/<reportID>/inputcontrols/<inputControlID>
```

> **Note**
>
> Since 4.2 SP04, a new '`allowNullValueSelection`' boolean attribute has been added to the widget attributes list for 'ComboBox', 'ListBox', 'CheckBox', and 'RadioButtons'.

Response

Response type: `application/xml` or `application/json`

Example: Calendar

```
<inputcontrol>
    <id>R2.IF3</id>
    <name>Invoice Date (type:DateTime)</name>
    <calendar useCustom="false" operator="Equal"/>
    <assignedDataObject refId="DP2.DO17" kind="Expression"/>
    <assignedReportElements>
        <assignedReportElement refId="1" reference="R2.IF3.T0"/>
    </assignedReportElements>
</inputcontrol>
```

Example: Check Box

```
<inputcontrol>
    <id>R1.IF0</id>
    <name>Country</name>
    <checkBox allowAllValuesSelection="true" useCustom="false" numberOfLines="5"
operator="InList">
        <default>
            <value>France</value>
            <value>US</value>
        </default>
    </checkBox>
    <assignedDataObject refId="DP0.DO6" kind="Expression"/>
    <assignedReportElements>
        <assignedReportElement refId="15" reference="R1.IF0.T0"/>
    </assignedReportElements>
</inputcontrol>
```

Example: Combo Box

```
<inputcontrol>
    <id>R2.IF2</id>
    <name>var1_serviceline</name>
```

```
    <comboBox allowAllValuesSelection="true" useCustom="false"
operator="Equal"/>
    <assignedDataObject refId="L1" kind="Variable"/>
    <assignedReportElements>
        <assignedReportElement refId="1" reference="R2.IF2.T0"/>
    </assignedReportElements>
</inputcontrol>
```

Example: List Box

```
<inputcontrol>
    <id>R1.IF1</id>
    <name>Service Line</name>
    <listBox allowAllValuesSelection="true" useCustom="false" numberOfLines="5"
cardinality="Multiple" operator="InList">
        <default>
            <value>Accommodation</value>
            <value>Food &amp; Drinks</value>
        </default>
    </listBox>
    <assignedDataObject refId="DP0.DO4" kind="Expression"/>
    <assignedReportElements>
        <assignedReportElement refId="15" reference="R1.IF1.T0"/>
    </assignedReportElements>
</inputcontrol>
```

Example: Radio Button

```
<inputcontrol>
    <id>R1.IF2</id>
    <name>Year</name>
    <radioButtons allowAllValuesSelection="true" useCustom="false"
numberOfLines="5" operator="Equal"/>
    <assignedDataObject refId="DP0.DO18" kind="Expression"/>
    <assignedReportElements>
        <assignedReportElement refId="15" reference="R1.IF2.T0"/>
    </assignedReportElements>
</inputcontrol>
```

Example: Slider

```
<inputcontrol>
    <id>R1.IF3</id>
    <name>Revenue</name>
    <slider minValue="30000" maxValue="600000" increment="1000"
cardinality="Interval" operator="Between">
        <default>
            <value>30000</value>
            <value>600000</value>
        </default>
    </slider>
```

```
    <assignedDataObject refId="DP0.DO7" kind="Expression"/>
    <assignedReportElements>
        <assignedReportElement refId="15" reference="R1.IF3.T0"/>
    </assignedReportElements>
</inputcontrol>
```

Example: Spinner

```
<inputcontrol>
    <id>R1.IF5</id>
    <name>var2_revenue</name>
    <spinner minValue="10" maxValue="15" increment="1" operator="Equal">
        <default>
            <value>11</value>
        </default>
    </spinner>
    <assignedDataObject refId="L2" kind="Variable"/>
    <assignedReportElements>
        <assignedReportElement refId="20" reference="R1.IF5.T0"/>
    </assignedReportElements>
</inputcontrol>
```

Example: Text Field

```
<inputcontrol>
    <id>R2.IF1</id>
    <name>Year</name>
    <textField operator="Equal">
        <default>
            <value>FY93</value>
        </default>
    </textField>
    <assignedDataObject refId="DP0.DO18" kind="Expression"/>
    <assignedReportElements>
        <assignedReportElement refId="1" reference="R2.IF1.T1"/>
    </assignedReportElements>
</inputcontrol>
```

Example: Tree List

```
<inputcontrol>
    <id>R2.IF4</id>
    <name>Customer Geography</name>
    <treeList numberOfLines="5" cardinality="Multiple"
allowComplexSelection="true" operator="InList"/>
    <assignedDataObject refId="DP1.DO9d" kind="Expression"/>
    <assignedReportElements>
        <assignedReportElement refId="1" reference="R2.IF4.T0"/>
    </assignedReportElements>
</inputcontrol>
```

Example: Since 4.2 SP04, the data type of the assigned data object is also returned in the output

```
GET /documents/6948/reports/1/inputcontrols/R1.IF0
```

Response:

```
<inputcontrol>
    <id>R1.IF0</id>
    <name>Invoice Date on CheckBox</name>
    <checkBox allowAllValuesSelection="true" useCustom="false" numberOfLines="5"
operator="InList">
        <default>
            <value>1992-09-03T20:33:00.000Z</value>
            <value>1992-10-13T21:34:00.000Z</value>
            <value>1993-01-11T00:00:00.000Z</value>
            <value>1993-02-01T18:30:00.000Z</value>
        </default>
    </checkBox>
    <assignedDataObject refId="DP0.DO17" kind="Expression"
dataType="DateTime"/>
    <assignedReportElements>
        <assignedReportElement refId="11" reference="R1.IF0.T0"/>
    </assignedReportElements>
</inputcontrol>
```

Example: Since 4.2 SP04, a new element related to grouping information has been added

```
GET /documents//8020/reports/48/inputcontrols/R48.IF2
```

Response:

```
<inputcontrol>
    <id>R48.IF2</id>
    ...
    <groupingInfo refId="R48.G0" inFilterPath="false" eligibility="Full"/>
</inputcontrol>
```

Example: With the 'eligibility' attribute

```
<inputcontrol>
    <id>R48.IF2</id>
    ...
    <groupingInfo refId="R48.G0" inFilterPath="false" eligibility="Full"/>
</inputcontrol>
```

Values for the `eligibility` attribute are:

- `None`
- `Restricted`

- `Full`

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the Input Controls of a Report [page 529]

8.5.2.4.1 Getting the Definition of an Input Control or a Group of Input Controls in a Report

Usage

Move eligible input controls into a group.

> ⓘ Note
>
> Since 4.2 SP4

Request

```
[GET] <url>/documents/{documentId}/reports/{reportId}/inputcontrols/
{inputcontrolId}
```

Parameters

`inputcontrolId`: a valid input control identifier

Response Type

Response type: `application/xml` or `application/json`

Example: XML Format

```
GET /documents/8020/reports/30/inputcontrols/R2.IF1
```

Response

```
<inputcontrol>
    <id>R2.IF1</id>
    <name>Quarter</name>
    <listBox allowAllValuesSelection="true" useCustom="false" numberOfLines="5"
cardinality="Multiple" operator="InList"/>
    <assignedDataObject refId="DP0.DOba" kind="Expression" dataType="String"/>
    <assignedReportElements>
        <assignedReportElement refId="17" reference="R2.IF1.T0"/>
    </assignedReportElements>
    <groupingInfo refId="R2.G0" inFilterPath="true" eligibility="Full"/>
</inputcontrol>
```

The new `<groupingInfo>` gives details about the grouping state (`refId` and `inFilterPath`: only present if the input control is attached to a group) and capabilities of the input control (eligibility: always present).

Values for the `eligibility` attribute are:

- `None`
- `Restricted`
- `Full`

Example: A group of input controls (XML format)

```
[GET] <url>/documents/8020/reports/30/inputcontrols/D.GO
```

Response

```
<inputcontrol>
    <id>D.G0</id>
    <name>Months</name>
    <group hasFilterPath="true">
        <id>D.IF0</id>
        <id>D.IF1</id>
    </group>
</inputcontrol>
```

8.5.2.5 Updating the Definition of an Input Control

Usage

Updates the details of an input control of a report.

Request

```
PUT /documents/<documentID>/reports/<reportID>/inputcontrols/<inputControlID>
```

> **Note**
>
> Since 4.2 SP04, a new '`allowNullValueSelection`' boolean attribute has been added to the widget attributes list for 'ComboBox', 'ListBox', 'CheckBox', and 'RadioButtons'.

The update of an input control must adhere to the following rules:

- You cannot change the widget of an input control.
- The input control must have at least one assigned report element.
- You can change the assigned data object, but it must be compatible with the widget.
- You can change the operator, but it must be compatible with the widget.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/7214/reports/1/inputcontrols/R1.IF2
```

Request body:

```
<inputcontrol>
    <name>IC Label</name>
    <radioButtons allowAllValuesSelection="true" useCustom="false"
numberOfLines="5" operator="NotEqual"/>
    <assignedDataObject refId="DP0.DO4"/>
    <assignedReportElements>
        <assignedReportElement refId="2"/>
    </assignedReportElements>
</inputcontrol>
```

Response:

```
<success>
    <message>The resource of type "InputControl" with identifier "R1.IF2" has
been uccessfully updated.</message>
    <id>R1.IF2</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the Input Controls of a Report [page 529]

8.5.2.5.1 Updating the Definition of an Input Control or Group of Input Controls in a Report

Usage

Update the definition of an existing regular input control or group of input controls.

For a regular input control attached to a group, in addition to the already known operations, it is also possible to remove it from the filter path.

> ⓘ Note
>
> Since 4.2 SP4

Request

```
[PUT] <url>/documents/{documentId}/report/{reportId}/inputcontrols/{inputcontrolId}
```

Parameters

`inputcontrolId`: a valid input control identifier

Content Type

Content type: `application/xml` or `application/json`

Response Type

Response type: `application/xml` or `application/json`

Example: Remove from Filter Path an Input Control Attached to a Group (XML format)

```
[PUT] <url>/documents/8020/reports/30/inputcontrols/R30.IF1
```

Request

```
<inputcontrol>
        <groupingInfo inFilterPath="false"/>
</inputcontrol>
```

Response

```
<success>
    <message>The resource of type "Input Control" with identifier "R30.IF1" has
been successfully updated.</message>
    <id>R30.IF1</id>
</success>
```

Example: Change the Name of a Group of Input Controls (XML format)

Groups of input controls may also be updated:

- To change their name
- To change the input controls list
- To reset the whole filter path.

```
[PUT] <url>/documents/8020/reports/1/inputcontrols/R1.G0
```

> **Note**
>
> The operations in the examples below may be combined in a single request.

Request

```
<inputcontrol>
        <name>New group name</name>
</inputcontrol>
```

Response

```
<success>
    <message>The resource of type "Input Control" with identifier "R1.G0" has
been successfully updated.</message>
    <id>R1.G0</id>
</success>
```

Example: Change the Regular Input Controls Attached to this Group (XML format)

```
[PUT] <url>/documents/8020/reports/1/inputcontrols/R1.G0
```

Request

```
<inputcontrol>
    <group>
        <id>R1.IF2</id>
        <id>R1.IF3</id>
    </group>
</inputcontrol>
```

Response

```
<success>
    <message>The resource of type "Input Control" with identifier "R1.G0" has
been successfully updated.</message>
    <id>R1.G0</id>
</success>
```

Example: Reset the Filter Path (XML format)

```
[PUT] <url>/documents/8020/reports/30/inputcontrols/R30.G3
```

Request

```
<inputcontrol>
    <group hasFilterPath='false'/>
</inputcontrol>
```

Response

```
<success>
    <message>The resource of type "Input Control" with identifier "R30.G3" has
been successfully updated.</message>
    <id>R30.G3</id>
</success>
```

8.5.2.6 Adding an Input Control

Usage

Adds a new input control to a report.

Request

`POST /documents/<documentID>/reports/<reportID>/inputcontrols`

The request body must adhere to the following rules:

- The widget must be compatible with the assigned data object, for example: slide on a measure, radio buttons on a dimension.
- The input control must have at least one assigned report element.
- The widget must be compatible with the operator.

> **Note**
>
> Since 4.2 SP04, a new `allowNullValueSelection` boolean attribute has been added to the widget attributes list for 'ComboBox', 'ListBox', 'CheckBox', and 'RadioButtons'.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`POST /documents/7216/reports/3/inputcontrols`

Request body:

```
<inputcontrol>
    <name>Slider on Revenue</name>
    <slider minValue="30000" maxValue="600000" increment="1000"
cardinality="Single" operator="Greater"/>
    <assignedDataObject refId="DP0.DO7"/>
    <assignedReportElements>
        <assignedReportElement refId="2"/>
    </assignedReportElements>
</inputcontrol>
```

Response:

```
<success>
    <message>The resource of type "InputControl" with identifier "R3.IF6" has
been successfully created.</message>
    <id>R3.IF6</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

8.5.2.7 Deleting an Input Control

Usage

Removes an input control from a report.

Request

```
DELETE /documents/<documentID>/reports/<reportID>/inputcontrols/<inputControlID>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/7232/reports/2/inputcontrols/R2.IF5
```

Response:

```
<success>
    <message>The resource of type "InputControl" with identifier "R2.IF5" has
been successfully removed.</message>
    <id>R2.IF5</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the Input Controls of a Report [page 529]

8.5.2.8 Getting the Selection of an Input Control

Usage

Gets the values selected for an input control of a report.

Request

```
GET /documents/<documentID>/reports/<reportID>/inputcontrols/<inputControlID>/
selection
```

Response

Response type: `application/xml` or `application/json`

The response is the list of values selected for the input control.

Example

```
GET /documents/7246/reports/1/inputcontrols/R1.IF4/selection
```

Response:

```
<selection>
    <value>Japan</value>
    <value>UK</value>
    <value>Australia</value>
</selection>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the Input Controls of a Report [page 529]

8.5.2.9 Setting the Selection of an Input Control

Usage

Sets the values selected for an input control of a report.

Request

```
PUT /documents/<documentID>/reports/<reportID>/inputcontrols/<inputControlID>/
selection
```

Request body: the values can be obtained by getting the list of values from the assigned data object.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/7246/reports/1/inputcontrols/R1.IF4/selection
```

Request body:

```
<selection>
    <value>Japan</value>
    <value>UK</value>
    <value>Australia</value>
</selection>
```

Response:

```
<success>
    <message>The resource of type "Input Control Selection" with identifier
"R1.IF4" has been successfully updated.</message>
    <id>R1.IF4</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]

Getting the Input Controls of a Report [page 529]
Getting the List of Values of a Data Object [page 596]

8.5.2.10 Deleting the Selection of an Input Control

Usage

Removes the values selected of an input control of a report.

Request

```
DELETE /documents/<documentID>/reports/<reportID>/inputcontrols/<inputControlID>/
selection
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/7255/reports/1/inputcontrols/R1.IF4/selection
```

Response:

```
<success>
    <message>The resource of type "InputControlSelection" has been successfully
removed.</message>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Reports [page 341]
Getting the Input Controls of a Report [page 529]

8.5.2.11 Reordering Input Controls in a Report

Usage

Changes the internal input control indexes to reorder them.

Request

```
PUT /documents/<documentId>/reports/<reportId>/inputcontrols
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example (XML)

```
PUT /documents/7070/reports/1/inputcontrols
```

Request:

```
<inputcontrols>
    <inputcontrol><id>R1.IF4</id></inputcontrol>
    <inputcontrol><id>R1.IF5</id></inputcontrol>
    <inputcontrol><id>R1.IF2</id></inputcontrol>
    <inputcontrol><id>R1.IF3</id></inputcontrol>
    <inputcontrol><id>R1.IF1</id></inputcontrol>
    <inputcontrol><id>R1.IF0</id></inputcontrol>
</inputcontrols>
```

Response:

```
<success>
    <message>The resource of type "Report" with identifier "1" has been
successfully updated.</message>
    <id>1</id>
</success>
```

8.5.2.12 Moving an Input Control from a Document to a Report

Usage

Moves an input control from a document to a report. The input control obtains a new identifier after the move, but keeps the same data objects.

You can also specify the report elements to associate with the input control in the request body.

Request

```
PUT /documents/<documentID>/reports/<reportID>/inputcontrols?fromId=<fromID>
```

Where:

- `<fromID>` is the input control ID in the document

Request body:

```
<inputcontrol>
    <assignedReportElements>
        <assignedReportElement refId="string"/>
```

The request body is optional.

You can retrieve the report element identifiers (`refId`) from the list of report elements (`<id>`):

```
GET documents/<documentID>/reports/<reportID>/elements
```

Response:

```
<elements>
    <element>
        <id>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: Moving an Input Control without Assigned Report Element

```
PUT /documents/6671/reports/2/inputcontrols?fromId=D.IF0
```

Response:

```
<success>
    <message>The resource of type "Input Control" with identifier "D.IF0" has
been successfully moved.</message>
    <id>R2.IF1</id>
</success>
```

The `<id>` is the new identifier of the input control in the report.

Example: Moving an Input Control with Assigned Report Elements

```
PUT /documents/6671/reports/2/inputcontrols?fromId=D.IF0
```

Request body:

```
<inputcontrol>
    <assignedReportElements>
        <assignedReportElement refId="10"/>
        <assignedReportElement refId="20"/>
        <assignedReportElement refId="19"/>
    </assignedReportElements>
</inputcontrol>
```

Response:

```
<success>
    <message>The resource of type "Input Control" with identifier "D.IF0" has
been successfully moved.</message>
    <id>R2.IF1</id>
</success>
```

Another call to retrieve the definition of the report input control shows the assigned report elements:

```
GET /documents/6671/reports/2/inputcontrols/R2.IF1
```

Response:

```
<inputcontrol>
    <id>R2.IF1</id>
    <name>Country</name>
    <comboBox allowAllValuesSelection="true" useCustom="false"
operator="Equal"/>
    <assignedDataObject refId="DP0.DO39" kind="Expression"/>
    <assignedReportElements>
        <assignedReportElement refId="10" reference="R2.IF1.T0"/>
        <assignedReportElement refId="20" reference="R2.IF1.T1"/>
        <assignedReportElement refId="19" reference="R2.IF1.T2"/>
    </assignedReportElements>
</inputcontrol>
```

8.5.2.13 Adding a New Group of Input Controls to a Report

Usage

Move eligible input controls into a group.

> **Note**
>
> Since 4.2 SP4

Request

```
[POST] <url>/documents/{documentId}/inputcontrols
```

Content Type

Content type: `application/xml` or `application/json`

Response Type

Response type: `application/xml` or `application/json`

Example: XML Format

```
POST /documents/8020/reports/30/inputcontrols
```

Request

```
<inputcontrol>
    <name>New group</name>
    <group>
        <id>R30.IF1</id>
        <id>R30.IF4</id>
        <id>R30.IF3</id>
    </group>
</inputcontrol>
```

Response

```
<success>
    <message>The resource of type "Input Control" with identifier "R30.G2" has
been successfully created.</message>
    <id>R30.G2</id>
</success>
```

8.5.2.14 Deleting a Group of Input Controls from a Report

Usage

This call removes a group of input controls (without deleting these regular input controls).

> **Note**
>
> Since 4.2 SP4

Request

```
[DELETE] <url>/documents/{documentId}/reports/reportId/inputcontrols/
{inputcontrolId}
```

Parameters

`inputcontrolId`: a valid input control identifier

Response Type

Response type: `application/xml` or `application/json`

Example: XML format

```
[DELETE] <url>/documents/8020/reports/2/D.G0
```

Response

```
<success>
    <message>The resource of type "Input Control" with identifier "D.G0" has
been successfully removed.</message>
    <id>D.G0</id>
</success>
```

8.6 Managing Data Providers

Data providers are data sources used to build queries in Web Intelligence documents.

A data provider can be:

- A universe
- A BEx query
- A Microsoft Excel 2003 or Microsoft Excel 2007 file
- A free-hand SQL script

> **Note**
>
> Multiflow free-hand SQL scripts are not supported in this release.

The present guide also describes methods specific to:

- Personal data providers based on Microsoft Excel files, see Managing Personal Data Providers [page 600]
- Free-hand SQL data providers, see Managing Connections for Free-Hand SQL Data Providers [page 616]

> **Restriction**
>
> The following workflows are not supported:
>
> - Using Analysis Views as data source for a new data provider
> - Changing data source when based on Analysis Views and Custom Data Provider

Getting the List of Data Providers [page 555]

Getting the Details of a Data Provider [page 557]

Adding a Data Provider [page 560]

Updating a Data Provider [page 563]

Moving a Data Provider in a Document [page 571]

Deleting a Data Provider [page 572]

Changing the Data Providers [page 573]

Getting the Flow Count of a Data Provider [page 583]

Getting the Details of a Flow [page 584]

Getting the Samples of a Flow [page 586]

Getting the Query Plan [page 587]

Updating the Query Plan [page 590]
Getting the Query Specification [page 592]
Updating the Query Specification [page 593]
Getting the Definition of an Object [page 595]
Getting the List of Values of a Data Object [page 596]
Getting the List of Values Under a Specific Data Object Value [page 598]

8.6.1 Getting the List of Data Providers

Usage

Gets the list of data providers of a Web Intelligence document.

The list can contain universes (`unv` and `unx`), BEx queries, Microsoft Excel files, free-hand SQL data providers, and Web Intelligence documents.

Request

```
GET /documents/<documentID>/dataproviders
```

Response

Response type: `application/xml` or `application/json`

Response body: the list of `<dataprovider>` elements with the following information:

- `<id>`
- `<name>`
- `<dataSourceId>` is the data source identifier
- `<dataSourceType>` is the type of data source (`unx`, `unv`, `bex`, `excel`, `fhsql`, `webi`)
- `<updated>` is the date of the last update

Example

```
GET /documents/7738/dataproviders
```

XML response:

```
<dataproviders>
```

```
    <dataprovider>
        <id>DP0</id>
        <name>Query 1</name>
        <dataSourceId>6187</dataSourceId>
        <dataSourceType>unv</dataSourceType>
        <updated>2006-09-20Z</updated>
    </dataprovider>
    <dataprovider>
        <id>DP1</id>
         <name>Query 2</name>
         <dataSourceId>6191</dataSourceId>
         <dataSourceType>unx</dataSourceType>
         <updated>2014-04-23T09:08:20.000+02:00</updated>
    </dataprovider>
    <dataprovider>
        <id>DP2</id>
        <name>Query 3</name>
        <dataSourceId>6120</dataSourceId>
        <dataSourceType>bex</dataSourceType>
        <updated>2014-04-23T09:08:20.000+02:00</updated>
    </dataprovider>
    <dataprovider>
        <id>DP3</id>
        <name>Query 4</name>
        <dataSourceId>6641</dataSourceId>
        <dataSourceType>excel</dataSourceType>
        <updated>2014-04-29T13:37:24.000+02:00</updated>
    </dataprovider>
    <dataprovider>
        <id>DP4</id>
        <name>Query 5</name>
        <dataSourceId>6250</dataSourceId>
        <dataSourceType>fhsql</dataSourceType>
        <updated>2014-08-29T08:32:37.000+02:00</updated>
    </dataprovider>
</dataproviders>
```

JSON response:

```
{"dataproviders":
    {"dataprovider":
        [{"id":"DP0","name":"Query 1","dataSourceId":6187,"dataSourceType":"unv",
            "updated":"2014-04-23T09:08:20.000+02:00"},
         {"id":"DP1","name":"Query 2","dataSourceId":6191,"dataSourceType":"unx",
            "updated":"2014-04-23T09:08:20.000+02:00"},
         {"id":"DP2","name":"Query 3","dataSourceId":6120,"dataSourceType":"bex",
            "updated":"2014-04-23T09:08:20.000+02:00"},
         {"id":"DP3","name":"Query
4","dataSourceId":6641,"dataSourceType":"excel",
            "updated":"2014-04-29T13:37:24.000+02:00"},
         {"id":"DP4","name":"Query
5","dataSourceId":6250,"dataSourceType":"fhsql",
            "updated":"2014-08-29T08:32:37.000+02:00"}]
    }
}
```

Related Information

Getting the List of Documents [page 226]

8.6.2 Getting the Details of a Data Provider

Usage

Gets the details of a data provider for a Web Intelligence document.

Request

```
GET /documents/<documentID>/dataproviders/<dataProviderID>
```

Response

Response type: `application/xml` or `application/json`

Response body: the details of the data provider, with:

- `<id>`
- `<name>`
- `<dataSourceId>`
- `<dataSourceType>` (`unx`, `unv`, `bex`, `excel`, `fhsql`, or `webi`)
- `<dataSourcePrefix>`, used as prefix for data source object IDs of `.unv` universes only
- `<updated>`, the date and time of the last update
- `<ispartial>`
- `<rowCount>`, the number of rows
- The object dictionary, with:
  - `<id>`
  - `<name>`
  - `<description>`
  - `<dataSourceObjectId>`
  - `<formulaLanguageId>`, used as key for data filters
  - `dataType`, `qualification`, and `highPrecision` attributes of `expression`
- The query identifier
- The properties of the data provider if any

> **Note**
>
> `highPrecision` is used in `Measure` expression definition to display the measure value with the "decimal floating-point" numeric format. This format, which is defined by the IEEE 754-2008 standard, allows the number of significant digits to go from 15 to 40, thus giving a higher precision to the value. The attribute default value is `false`.

Example

```
GET /documents/18809/dataproviders/DP0
```

XML response:


```
<dataprovider>
    <id>DP0</id>
    <name>Query 1</name>
    <dataSourceId>7599</dataSourceId>
    <dataSourceType>bex</dataSourceType>
    <dataSourcePrefix>DS0</dataSourcePrefix>
    <updated>2012-07-31T15:44:25.000+02:00</updated>
    <duration>1</duration>
    <isPartial>false</isPartial>
    <rowCount>284</rowCount>
    <flowCount>1</flowCount>
    <dictionary>
        <expression dataType="String" qualification="Dimension">
            <id>DP1.DO1</id>
            <name>City</name>
            <description>City</description>
            <dataSourceObjectId>AZ_CITY</dataSourceObjectId>
            <formulaLanguageId>[City]</formulaLanguageId>
        </expression>
        <expression dataType="String" qualification="Dimension">
            <id>DP1.DO1d</id>
            <name>Customer</name>
            <description>Customer</description>
            <dataSourceObjectId>AZ_CUSTOM</dataSourceObjectId>
            <formulaLanguageId>[Query 2].[Customer]</formulaLanguageId>
        </expression>
        <expression dataType="String" qualification="Dimension">
            <id>DP1.DO6</id>
            <name>Sales Person</name>
            <description>Sales Person</description>
            <dataSourceObjectId>AZ_SALES</dataSourceObjectId>
            <formulaLanguageId>[Query 2].[Sales Person]</formulaLanguageId>
        </expression>
        <expression dataType="Numeric" qualification="Measure"
highPrecision="true">
            <id>DP1.DO5c</id>
            <name>Order Amount</name>
            <description>Order Amount</description>
            <dataSourceObjectId>MCWZZCK8Z39ES5HKKFA52FG78E</dataSourceObjectId>
            <formulaLanguageId>[Query 2].[Order Amount]</formulaLanguageId>
        </expression>
        <expression dataType="Numeric" qualification="Measure"
highPrecision="false">
            <id>DP1.DO5f</id>
            <name>Order Quantity</name>
            <description>Order Quantity</description>
            <dataSourceObjectId>M7MRKCICEBY5WYKQ8AL86VY9Q8</dataSourceObjectId>
            <formulaLanguageId>[Query 2].[Order Quantity]</formulaLanguageId>
        </expression>
    </dictionary>
    <query>CgASFQoGWl9DSVRZEgsKB0FaX0NJVFkQARIXCgdaX1NBTEVTEgwKCEFaX1NBTEVTEAES
        GQoIWl9DVVNUT00SDQoJQVpfQ1VTVE9NEAEaSwoZQ1daWkNLOFozOUVTNUhLS0ZBNTJGRzc
        4RRIMT3JkZXIgQW1vdW50GAIiHgoaTUNXWlpDSzhaMzlFUzVIS0tGQTUyRkc3OEUQBhpNCh
        k3TVJLQ0lDRUJZNVdZS1E4QUw4NlZZOVE4Eg5PcmRlciBRdWFudGl0eRgCIh4KGk03TVJLQ
        0lDRUJZNVdZS1E4QUw4NlZZOVE4EAYgACgAMAE=</query>
</dataprovider>
```


JSON response:


```
{"dataprovider":
```

```
    {"id":"DP2","name":"Query 3","dataSourceId":6120,"dataSourceType":"bex",
        "dataSourcePrefix":"DS0", "updated":"2014-04-23T09:08:20.000+02:00",
        "duration":1,"isPartial":false, "rowCount":175,"flowCount":1,
        "dictionary":
            {"expression":

[{"@dataType":"String","@qualification":"Hierarchy","id":"DP2.DO5",
                    "name":"Region","description":"Region","dataSourceObjectId":
                    "HZ_REGION", "formulaLanguageId":"[Region]"},

{"@dataType":"String","@qualification":"Attribute","id":"DP2.DOa",
                    "name":"Order Amount Currency","description":"",
                    "dataSourceObjectId":"MDCZO1XT12V8KC6LFFGN1WEL3E.Currency",
                    "formulaLanguageId":"[Order Amount Currency]"}
                ]
            },

"query":"CgASGQoIWl9SRUdJT04SDQoJSFpfUkVHSU9OEAIaVAoZRENaTzFYVDEyVjhLQzZMRkZ

HTjFXRUwzRRIMT3JkZXIgQW1vdW50GAIiJwojTURDWk8xWFQxMlY4S0M2TEZGR04xV0VMM0U
            uQ3VycmVuY3kQCCAAKAAwAA=="
    }
}
```


XML response in the case of a Microsoft Excel file:


```
<dataprovider>
    <id>DP3</id>
    <name>MyQuery</name>
    <dataSourceId>6641</dataSourceId>
    <dataSourceType>excel</dataSourceType>
    <updated>2014-04-29T13:37:24.000+02:00</updated>
    <duration>1</duration>
    <isPartial>false</isPartial>
    <rowCount>405</rowCount>
    <dictionary>
        <expression dataType="String" qualification="Dimension">
            <id>DP3.DO7</id>
            <name>Build</name>
            <description/>
            <dataSourceObjectId>DS3.DO7</dataSourceObjectId>
            <formulaLanguageId>[Build]</formulaLanguageId>
        </expression>
        <expression dataType="String" qualification="Dimension">
            <id>DP3.DO0</id>
        ...
    </dictionary>
    <properties>
        <property key="selectedSheet">Planning</property>
        <property key="rangeSelectionMode">all</property>
        <property key="firstRawAsObjectNames">true</property>
        <property key="isRefreshable">true</property>
    </properties>
</dataprovider>
```


XML response in the case of a free-hand SQL script:


```
<dataprovider>
    <id>DP5</id>
    <name>FreeHandSQL based on beachOraJdbc</name>
    <dataSourceId>6250</dataSourceId>
    <dataSourceType>fhsql</dataSourceType>
    <updated>2014-08-29T08:43:26.000+02:00</updated>
    <duration>1</duration>
    <isPartial>false</isPartial>
    <rowCount>7</rowCount>
```

```
    <flowCount>1</flowCount>
    <dictionary>
        <expression dataType="Numeric" qualification="Measure"
highPrecision="false">
            <id>DP5.DO0</id>
            <name>COUNTRY_ID</name>
            <dataSourceObjectId>DS5.DO0</dataSourceObjectId>
            <formulaLanguageId>[COUNTRY_ID]</formulaLanguageId>
            <aggregationFunction>None</aggregationFunction>
        </expression>
        <expression dataType="String" qualification="Dimension">
            <id>DP5.DO1</id>
            <name>COUNTRY</name>
            <dataSourceObjectId>DS5.DO1</dataSourceObjectId>
            <formulaLanguageId>[COUNTRY]</formulaLanguageId>
        </expression>
    </dictionary>
    <properties>
        <property key="sql">SELECT * from country</property>
        <property key="maxRows">-1</property>
        <property key="timeout">-1</property>
    </properties>
</dataprovider>
```

Related Information

Managing Data Filters in Reports [page 365]
Getting the List of Data Providers [page 555]

8.6.3 Adding a Data Provider

Usage

Adds a new data provider to a Web Intelligence document.

You can add:

- A universe (unv or unx)
- A BEx query
- A Microsoft Excel 2003 or Microsoft Excel 2007 file
- A free-hand SQL script

Request

`POST /documents/<documentID>/dataproviders`

Request body:

```
<dataprovider>
```

```
    <name>
    <dataSourceId>
```

Where:

- `<name>` is the data source name
- `<dataSourceId>` is the data source identifier

Request Body for a Microsoft Excel File

In the case of a Microsoft Excel file, you may add the following `<property>` elements to provide supplementary details on the data source.

```
<properties>
    <property key="selectedSheet">
    <property key="rangeSelectionMode">
    <property key="firstRowAsObjectNames">
    <property key="selectedRange">
```

Properties

Property Key	Description
`selectedSheet`	Specifies the name of the sheet to be used as data provider. The first sheet is used if the property is not set.
`rangeSelectionMode`	Can be: • `all`, if the whole sheet is selected • `named`, if one of the named cell ranges is selected • `freehand`, if the end-user selects the cell range
`firstRowAsObjectNames`	Boolean. Default is `true`. If `true`, the first row of the sheet contains column names.
`selectedRange`	Do not use it if `rangeSelectionMode` is set to `all`. Specify one of the named cell ranges if any, or use the end-user input. **Note** The present release does not suppot fragmented named ranges.

Request body for a Free-Hand SQL Script

In the case of a free-hand SQL script, the request body must contain a SQL statement as `<property>`:

```
<dataprovider>
    <name>
    <dataSourceId>
    <properties>
        <property key="sql">
```

All SQL statements conform to standards ANSI-SQL 89/92/98 are supported. For example, SET options, stored procedures, and WITH clauses are supported. The @Variable and @Prompt functions for interactive

queries are also supported. A SQL script containing DDL commands such as DROP TABLE or ALTER TABLE is not executed. Multiflow SQL statements are not supported. The query will return the first result only.

You may also add the following `<property>` elements to provide supplementary details on the desired results:

```
<properties>
    <property key="maxRows">
    <property key="timeout">
```

Properties

Property Key	Description
`maxRows`	Specifies the maximum rows retrieved. If not set or value is `-1`, the property is disabled.
`timeout`	Specifies the maximum retrieval time (in seconds). If not set or value is `-1`, the property is disabled.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
POST /documents/3422/dataproviders
```

Request body for a BEx query:

```
<dataprovider>
    <name>Query1</name>
    <dataSourceId>11990;Z_BOBJ;AAQUERY_SAMPLE</dataSourceId>
<dataprovider>
```

Request body for a Microsoft Excel file:

```
<dataprovider>
    <name>My Excel data provider</name>
    <dataSourceId>6641</dataSourceId>
    <properties>
        <property key="selectedSheet">mysheet</property>
        <property key="rangeSelectionMode">freehand</property>
        <property key="firstRowAsObjectNames">true</property>
        <property key="selectedRange">A2:F7</property>
    </properties>
<dataprovider>
```

Request body for a free-hand SQL script:

```
<dataprovider>
    <name>FHSQL dp on beachOraJdbc</name>
    <dataSourceId>6250</dataSourceId>
    <properties>
        <property key="sql">SELECT * from country</property>
```

```
        <property key='maxRows'>286</property>
        <property key='timeout'>857</property>
    </properties>
</dataprovider>
```

Response:

```
<success>
    <message>The resource of type "Data provider" with identifier "DP3" has been
successfully created.</message>
    <id>DP3</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the Details of a Microsoft Excel File [page 603]

8.6.4 Updating a Data Provider

Usage

Purges or updates a data provider.

The following table shows what you can update in the particular cases of a data provider based on a Microsoft Excel file or a free-hand SQL script.

Data Provider	Possible Updates

Microsoft Excel file	• The Microsoft Excel file itself
	• Properties
	• Dictionary expression details:
	  • Data types
	  • Qualification
	  • High Precision
	  • Name
	  • Aggregation functions for measures
	  • Associated dimensions for attributes

Data Provider	Possible Updates

Free-hand SQL script	• Properties
	• Connections
	• SQL script
	• Dictionary expression details:
	  • Qualification
	  • High Precision
	  • Name
	  • Aggregation functions for measures
	  • Associated dimensions for attributes

> **Note**
>
> - The replacing Microsoft Excel file must be stored in the CMS repository. Also, it must have the same file structure as the one used as data provider (column number, names, and order).
> - The Microsoft Excel file and its properties can be changed through either one request or two distinct requests.
> - The connection and SLQ script can be changed through either one request or two distinct requests.
> - Changes to `<dataSourceObjectId>` and `<formulaLanguageId>` are ignored during update.
> - The SDK handles missing result objects, duplicate expression identifier, duplicate name, unknown expression identifier, and modified object order.

Request

```
PUT /documents/<documentID>/dataproviders/<dataProviderID>?
purge=<purge>&purgeOptions=<purgeOptions>
```

Where:

- `<purge>` (optional) indicates whether to purge the data provider or not. Default is `false`.
- `<purgeOptions>` (optional) is `prompts` to control the purge operation. If not set or empty, only the data provider will be purged.

> **Note**
>
> An error occurs when `purge=false` and `purgeOptions` are both specified in the call.

The request body is optional.

Request Body to Change the Name of a Data Provider

```
<dataprovider>
    <name>
```

Request Body to Change the Format of a Measure Value of a Data Provider

You can set the `highPrecision` attribute of an expression of type measure to `true` to display the measure value with the "decimal floating-point" numeric format. This format, which is defined by the IEEE 754-2008 standard, allows the number of significant digits to go from 15 to 40, thus giving a higher precision to the value. You can make this change to any data provider.

```
<dataprovider>
    <dictionary>
        <expression highPrecision=Boolean>
            <id>
```

If the attribute is not in the request body, the measure is displayed as a usual numeric.

Request Body to Change Properties of a Microsoft Excel File

```
<dataprovider>
    <property key="selectedSheet">
    <property key="rangeSelectionMode">
    <property key="firstRowAsObjectNames">
    <property key="selectedRange">
```

You can also add the `isRefreshable` property to the request body to make the data provider refreshable:

```
<dataprovider>
    <property key="isRefreshable">true|false</property>
```

The `isRefreshable` property value has an impact on the purge operation. If this property is `false` on the Web Intelligence server, the purge is not performed even if `<purge>` is set to `true`.

Request Body to Change the Microsoft Excel File Used as Data Provider

```
<dataprovider>
    <dataSourceId>
```

Request Body to Change Properties of a Free-Hand SQL Script

```
<dataprovider>
    <property key="maxRows">
    <property key="timeout">
```

Request Body to Change the Connection and the SQL Script of a Free-Hand SQL Data Provider

```
<dataprovider>
    <dataSourceId>
    <properties>
        <property key="sql">
```

`<dataSourceId>` indicates the connection ID.

Connection and SQL script can be updated in separate requests.

Errors are thrown if:

- Connection and SQL script are not compatible (HTTP error code 400, WSR 00102).
- The end-user is not allowed to edit the SQL script (HTTP error code 401, WSR 00402).

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: To Purge a Data Provider, but not Prompts

`PUT /documents/7738/dataproviders/DP0?purge=true`

```
<success>
    <message>The resource of type "Data provider" with identifier "DP0" has been
successfully updated.</message>
    <id>DP0</id>
</success>
```

Example: To Purge a Data Provider and Prompts

`PUT /documents/7738/dataproviders/DP0?purge=true&purgeOptions=prompts`

```
<success>
    <message>The resource of type "Data provider" with identifier "DP0" has been
successfully updated.</message>
    <id>DP0</id>
</success>
```

Example: To Rename a Data Provider

`PUT /documents/7738/dataproviders/DP0`

Request body in an XML file:

```
<dataprovider>
    <name>My Renamed Data Provider</name>
</dataprovider>
```

Response:

```
<success>
    <message>The resource of type "Data provider" with identifier "DP0" has been
successfully updated.</message>
    <id>DP0</id>
</success>
```

Example: To Modify the Free-Hand SQL Data Provider Properties

`PUT /documents/8229/dataproviders/DP0`

Request body in an XML file:

```
<dataprovider>
    <property key="maxRows">928</property>
    <property key="timeout">101</property>
</dataprovider>
```

Response:

```
<success>
    <message>The resource of type "Data provider" with identifier "DP0" has been
successfully updated.</message>
    <id>DP0</id>
</success>
```

Example: To Change the Connection and SLQ Script of a Free-Hand SQL Data Provider

`PUT /documents/8229/dataproviders/DP0`

Request body:

```
<dataprovider>
    <dataSourceId>6340</dataSourceId>
    <properties>
        <property key="sql">SELECT DISTINCT SALES.INVOICE_DATE FROM SALES</
property>
    </properties>
</dataprovider>
```

Response:

```
<success>
    <message>The resource of type "Data provider" with identifier "DP0" has been
successfully updated.</message>
    <id>DP0</id>
<success>
```

Example: To Update the Object Dictionary of a Data Provider Based on a Microsoft Excel File

A first call allows you to get the data provider details:

`GET /documents/11837/dataproviders/DP0`

Response:

```
<dataprovider>
    <id>DP0</id>
    <name>named cell</name>
    <dataSourceId>11835</dataSourceId>
    <dataSourceType>excel</dataSourceType>
    <duration>0</duration>
    <isPartial>false</isPartial>
    <rowCount>0</rowCount>
    <flowCount>1</flowCount>
    <dictionary>
        <expression dataType="String" qualification="Dimension">
            <id>DP0.DO0</id>
            <name>OrderDate</name>
            <dataSourceObjectId>DS0.DO0</dataSourceObjectId>
            <formulaLanguageId>[OrderDate]</formulaLanguageId>
        </expression>
        <expression dataType="String" qualification="Dimension">
            <id>DP0.DO1</id>
            <name>Region</name>
            <dataSourceObjectId>DS0.DO1</dataSourceObjectId>
            <formulaLanguageId>[Region]</formulaLanguageId>
        </expression>
        <expression dataType="String" qualification="Dimension">
            <id>DP0.DO2</id>
            <name>Rep</name>
            <dataSourceObjectId>DS0.DO2</dataSourceObjectId>
            <formulaLanguageId>[Rep]</formulaLanguageId>
        </expression>
        <expression dataType="String" qualification="Dimension">
            <id>DP0.DO3</id>
            <name>Item</name>
            <dataSourceObjectId>DS0.DO3</dataSourceObjectId>
            <formulaLanguageId>[Item]</formulaLanguageId>
        </expression>
        <expression dataType="Numeric" qualification="Measure">
            <id>DP0.DO4</id>
            <name>Units</name>
            <dataSourceObjectId>DS0.DO4</dataSourceObjectId>
            <formulaLanguageId>[Units]</formulaLanguageId>
            <aggregationFunction>Sum</aggregationFunction>
        </expression>
        <expression dataType="Numeric" qualification="Measure">
            <id>DP0.DO5</id>
            <name>Cost</name>
```

```
            <dataSourceObjectId>DS0.DO5</dataSourceObjectId>
            <formulaLanguageId>[Cost]</formulaLanguageId>
            <aggregationFunction>Sum</aggregationFunction>
        </expression>
    </dictionary>
    <properties>
        <property key="selectedSheet">Sheet1</property>
        <property key="rangeSelectionMode">freehand</property>
        <property key="selectedRange">A1:F9</property>
        <property key="firstRowAsObjectNames">true</property>
        <property key="isRefreshable">true</property>
    </properties>
</dataprovider>
```

A second call allows you to change:

- The names of `DP0.DO0`, `DP0.DO1`, `DP0.DO2`, `DP0.DO3`, `DP0.DO4`, and `DP0.DO5`.
- The qualification of `DP0.DO2` and `DP0.DO5`
- The associated dimension of `DP0.DO2`
- The aggregation function of `DP0.DO4`

`PUT /documents/11837/dataproviders/DP0`

Request body:

```
<dataprovider>
    <dictionary>
        <expression dataType="String" qualification="Dimension">
            <id>DP0.DO0</id>
            <name>OrderDate1</name>
        </expression>
        <expression dataType="String" qualification="Dimension">
            <id>DP0.DO1</id>
            <name>Region1</name>
        </expression>
        <expression dataType="String" qualification="Attribute">
            <id>DP0.DO2</id>
            <name>Rep1</name>
            <associatedDimensionId>DP0.DO1</associatedDimensionId>
        </expression>
        <expression dataType="String" qualification="Dimension">
            <id>DP0.DO3</id>
            <name>Item1</name>
        </expression>
        <expression dataType="Numeric" qualification="Measure">
            <id>DP0.DO4</id>
            <name>Average</name>
            <aggregationFunction>Average</aggregationFunction>
        </expression>
        <expression dataType="Numeric" qualification="Dimension">
            <id>DP0.DO5</id>
            <name>Cost1</name>
        </expression>
    </dictionary>
</dataprovider>
```

Response:

```
<success>
    <message>The resource of type "Data provider" with identifier "DP0" has been
successfully updated.</message>
    <id>DP0</id>
</success>
```

In a third call, you check the dictionary is updated. `<formulaLanguageId>` have changed automatically depending on the `<name>` change.

`GET /documents/11837/dataproviders/DP0`

Response:

```
<dataprovider>
    <id>DP0</id>
    <name>named cell</name>
    <dataSourceId>11835</dataSourceId>
    <dataSourceType>excel</dataSourceType>
    <duration>0</duration>
    <isPartial>false</isPartial>
    <rowCount>0</rowCount>
    <flowCount>1</flowCount>
    <dictionary>
        <expression dataType="String" qualification="Dimension">
            <id>DP0.DO0</id>
            <name>OrderDate1</name>
            <dataSourceObjectId>DS0.DO0</dataSourceObjectId>
            <formulaLanguageId>[OrderDate1]</formulaLanguageId>
        </expression>
        <expression dataType="String" qualification="Dimension">
            <id>DP0.DO1</id>
            <name>Region1</name>
            <dataSourceObjectId>DS0.DO1</dataSourceObjectId>
            <formulaLanguageId>[Region1]</formulaLanguageId>
        </expression>
        <expression dataType="String" qualification="Attribute">
            <id>DP0.DO2</id>
            <name>Rep1</name>
            <dataSourceObjectId>DS0.DO2</dataSourceObjectId>
            <formulaLanguageId>[Rep1]</formulaLanguageId>
            <associatedDimensionId>DP0.DO1</associatedDimensionId>
        </expression>
        <expression dataType="String" qualification="Dimension">
            <id>DP0.DO3</id>
            <name>Item1</name>
            <dataSourceObjectId>DS0.DO3</dataSourceObjectId>
            <formulaLanguageId>[Item1]</formulaLanguageId>
        </expression>
        <expression dataType="Numeric" qualification="Measure">
            <id>DP0.DO4</id>
            <name>Average1</name>
            <dataSourceObjectId>DS0.DO4</dataSourceObjectId>
            <formulaLanguageId>[Average1]</formulaLanguageId>
            <aggregationFunction>Average</aggregationFunction>
        </expression>
        <expression dataType="Numeric" qualification="Dimension">
            <id>DP0.DO5</id>
            <name>Cost1</name>
            <dataSourceObjectId>DS0.DO5</dataSourceObjectId>
            <formulaLanguageId>[Cost1]</formulaLanguageId>
        </expression>
    </dictionary>
    <properties>
        <property key="selectedSheet">Sheet1</property>
        <property key="rangeSelectionMode">freehand</property>
        <property key="selectedRange">A1:F9</property>
        <property key="firstRowAsObjectNames">true</property>
        <property key="isRefreshable">true</property>
    </properties>
</dataprovider>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Data Providers [page 555]

8.6.5 Moving a Data Provider in a Document

Usage

Moves an existing data provider in a Web Intelligence document from a position to another.

Request

```
PUT /documents/<documentID>/dataproviders?
fromId=<FromDataProviderID>&toId=<ToDataProviderID>
```

Where:

- `<FromDataProviderID>` is the identifier of the data provider to move
- `<ToDataProviderID>` is the identifier of the data provider after move

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/8022/dataproviders?fromId=DP0&toId=DP2
```

```
<success>
    <message>Resources of type "Data provider" with identifier "DP0" has been
successfully moved.</message>
    <id>DP2</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Data Providers [page 555]

8.6.6 Deleting a Data Provider

Usage

Removes a data provider from the list of data providers of a Web Intelligence document.

Request

```
DELETE /documents/<documentID>/dataproviders/<dataProviderID>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/7738/dataproviders/DP1
```

```
<success>
    <message>The resource of type "Data provider" with identifier "DP1" has been
successfully removed.</message>
    <id>DP1</id>
</success>
```

Example: Removing the Last Data Provider

```
DELETE /documents/7738/dataproviders/DP0
```

```
<error>
   <error_code>101</error_code>
   <message>The resource of type "Data provider" with identifier "DP0" cannot be
removed.</message>
```

```
</error>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Data Providers [page 555]

8.6.7 Changing the Data Providers

You can change the data providers of a Web Intelligence document by replacing data objects from a current data source with data objects from a target data source.

> → Remember
>
> - Starting with the 4.2 SP3 release, the document parameters (prompt and contexts) do not need to be answered when changing data providers of a document. Only data source parameters such as SAP HANA variables and BEx query variables remain to be answered.
> - Before 4.2 SP3, the ID of a data provider affected by a "change source" was modified from N to N+1, for example from `DP0` to `DP1`. It is no longer the case since 4.2 SP3.

> ⓘ Note
>
> The "change source" functionality does not support text files, Microsoft Excel spreadsheets, SAP HANA Online and Web Services as data sources.

Getting the Possible Object Mappings Using the Default Strategies [page 573]

Getting the Possible Object Mappings Using Selected Strategies [page 576]

Changing the Data Objects of a Data Provider [page 579]

Related Information

Change Source [page 145]

8.6.7.1 Getting the Possible Object Mappings Using the Default Strategies

Usage

Gets the list of possible object mappings that have been found using the default strategies.

If no data provider is specified, the request is run for all the existing data providers in the document.

Request

```
GET /documents/<documentID>/dataproviders/mappings?
originDataproviderIds=<dataProviderID[,...]>&targetDatasourceId=<dataSourceID>
```

Where:

- `<dataProviderID[,...]>` is the list of data provider identifiers to consider for the mapping, separated by commas. This parameter is optional. If not present, all the data providers in the document are considered.
- `<dataSourceID>` is the target data source identifier to use for the proposal mapping. This parameter is mandatory.

Response

Response type: `application/xml` or `application/json`

Response body: the details of the mappings, described in Mapping Request and Response Body Schemas [page 147].

Example: To Get the Possible Mappings

Data provider IDs are DP0 and DP1. Target data source ID is 11414.

```
GET /documents/9460/dataproviders/mappings?
originDataproviderIds=DP0,DP1&targetDatasourceId=11414
```

```
<mappings>
    <content>
        <mapping status="Ok">
            <source>
                <id>DS0.DO12</id>
            </source>
            <target>
                <id>DS1.DO12</id>
            </target>
        </mapping>
        <mapping status="Ok">
            <source>
                <id>DS0.DO13</id>
            </source>
            <target>
                <id>DS1.DO13</id>
            </target>
        </mapping>
        ...
        <mapping status="Ok">
            <source>
```

```
                <id>DS0.DO7</id>
            </source>
            <target>
                <id>DS1.DO7</id>
            </target>
        </mapping>
        <mapping status="Ambiguous">
            <source>
                <id>DS0.DO84</id>
            </source>
            <target>
                <id>DS1.DO6</id>
            </target>
        </mapping>
    </content>
</mappings>
```

Example: To Get the Possible Mappings When the Target is a BEx Query

Data provider IDs are DP0 and DP1. Target data source ID is "10326;AAQUERY_RESTRICT_KF".

```
GET /documents/10554/dataproviders/mappings?
originDataproviderIds=DP0,DP1&targetDatasourceId=10326;AAQUERY_RESTRICT_KF
```

```
<mappings>
    <content>
        <mapping status="Ok">
            <source>
                <id>AZ_CITY</id>
            </source>
            <target>
                <id>AZ_CITY</id>
            </target>
        </mapping>
        <mapping status="Ambiguous">
            <source>
                <id>MA55OT46EO4K803Z77TO03LNMU</id>
            </source>
            <target>
                <id>MD4NTN2159JU9ONUUHTLMNBW3U</id>
            </target>
        </mapping>
    </content>
</mappings>
```

Related Information

Change Source [page 145]
Getting the List of Documents [page 226]
Getting the List of Data Providers [page 555]

8.6.7.2 Getting the Possible Object Mappings Using Selected Strategies

Usage

Gets the list of possible object mappings that have been found using the given strategies.

These strategies apply in the order specified in the request when the mapping status is "not found", until one object match is found.

If no data provider is specified, the request is run on all the existing data providers in the document.

Request

```
PUT /documents/<documentID>/dataproviders/mappings?
originDataproviderIds=<dataProviderID[,...]>&targetDatasourceId=<dataSourceID>
```

Where:

- `<dataProviderID[,...]>` is the list of data provider identifiers to consider for the mapping, separated by commas. This parameter is optional. If not present, all the data providers in the document are considered.
- `<dataSourceID>` is the target data source identifier to use for the proposal mapping. This parameter is mandatory.

Request body:

```
<mappings>
    <policy qualificationTolerance="Low|Normal|High" dataTypeTolerance="Low|
Normal|High">
        <strategies mappingSourceIds="string">
            <strategy name="SameId|SameName|SameTechnicalName|SamePath|CloseName|
Selection|Removal" targetId="string" />
    <content>
        <mapping>
            <source>
                <id>
            <target>
                <id>
    <parameters>
        <parameter>
```

See Mapping Request and Response Body Schemas [page 147] for request body description.

Response

Response type: `application/xml` or `application/json`

Response body: the details of the mappings, described in Mapping Request and Response Body Schemas [page 147].

Example

```
PUT /documents/7278/dataproviders/mappings?targetDatasourceId=6610
```

Request body:

```
{"mappings": {
    "policy": {
      "strategies": [
        {
          "strategy": [
            {"@name": "SamePath"},
            {"@name": "SameTechnicalName"},
            {"@name": "SameName"},
            {"@name": "Removal"}]
        },
        {
          "@mappingSourceIds": "DS0.DO2,DS0.DO7,DS0.DO18,DS0.DO4",
          "strategy": [
            {"@name": "SameId"},
            {"@name": "CloseName"}]
        },
        {
          "@mappingSourceIds": "DS0.DO25",
          "strategy": {
            "@name": "Selection",
            "@targetId": "M[Measures].[Customer Count]"
          }
        },
        {
          "@mappingSourceIds": "DS0.DO7",
          "strategy": {
            "@name": "Selection",
            "@targetId": "M[Measures].[Internet Sales Amount]"
          }
        }
      ]
    },
    "parameters": {
      "parameter": {
        "id": "1",
        "answer": {
          "values": {
            "value": {
              "@id": "2",
              "$": "France"
            }
          }
        }
      }
    }
  }
}
```

Response:

```
{"mappings": {
    "policy": {
      "strategies": [
        {
          "strategy": [
            {"@name":"SamePath"},
            {"@name":"SameTechnicalName"},
            {"@name":"SameName"},
            {"@name":"Removal"}]
```

```
        },
        {
          "@mappingSourceIds":"DS0.DO2,DS0.DO7, DS0.DO18, DS0.DO4",
          "strategy": [
            {"@name":"SameId"},
            {"@name":"CloseName"}]
        },
        {
          "@mappingSourceIds":"DS0.DO25",
          "strategy": {
            "@name":"Selection",
            "@targetId":"M[Measures].[Customer Count]"}
        },
        {
          "@mappingSourceIds":"DS0.DO7",
          "strategy": {
            "@name":"Selection",
            "@targetId":"M[Measures].[Internet Sales Amount]"}
        }
      ]
    },
    "content": {
      "mapping": [
        {
          "@status":"Ambiguous",
          "source": {"id":"DS0.DO12"},
          "target": {"id": "L[Sales Territory].[Sales Territory].[Region]"}
       },
        {
          "@status":"Ambiguous",
          "source": {"id":"DS0.DO13"},
          "target": {"id": "A[Customer].[City]"}
        },
        {
          "@status":"NotFound",
          "source": {"id":"DS0.DO17"}
        },
        {
          "@status":"Ambiguous",
          "source": {"id":"DS0.DO18"},
          "target": {"id":"A[Customer].[City]"}
        },
        {
          "@status":"Ambiguous",
          "source": {"id":"DS0.DO2"},
          "target": {"id":"A[Product].[Color]"}
        },
        {
          "@status":"Ambiguous",
          "source": {"id":"DS0.DO25"},
          "target": {"id":"M[Measures].[Customer Count]"}
        },
        {
          "@status":"Ambiguous",
          "source": {"id":"DS0.DO4"},
          "target": {"id":"L[Internet Sales Order Details].[Internet Sales
Orders].[Order Line]"}
        },
        {
          "@status":"Ambiguous",
          "source": {"id": "DS0.DO6"},
          "target": {"id":"A[Customer].[Country]"}
        },
        {
          "@status":"Ambiguous",
          "source": {"id":"DS0.DO7"},
          "target": {"id":"M[Measures].[Internet Sales Amount]"}
        },
```

```
        {
          "@status":"Ambiguous",
          "source": {"id":"DS0.DO84"},
          "target": {"id":"A[Customer].[Country]"}
        }
      ]
    }
  }
}
```

Related Information

Change Source [page 145]
Getting the List of Documents [page 226]
Getting the List of Data Providers [page 555]

8.6.7.3 Changing the Data Objects of a Data Provider

Usage

Replaces data objects of the current data source with the ones of the target data source using a given data object mapping and a given strategy. The use of custom strategies is new since the 4.1 SP6 release.

Starting with the 4.2 SP3 release, the document parameters (prompt and contexts) do not need to be answered when changing data providers of a document. Only data source parameters such as SAP HANA variables and BEx query variables remain to be answered.

Request

```
POST /documents/<documentID>/dataproviders/mappings?
originDataproviderIds=<dataProviderID[,...]>&targetDatasourceId=<dataSourceID>&skip
Checking=true
```

Where:

- `<dataProviderID[,...]>` is the list of data provider identifiers to consider for the mapping, separated by commas. This parameter is optional.
- `<dataSourceID>` is the target data source identifier to use for the proposal mapping. This parameter is mandatory.
- `skipChecking` allows to skip all the verifications to speed up the change source process. This parameter is optional.

Request body: the strategies, the custom mapping, and the parameter values if the data source has parameters. If the request body is empty, the change source will be done using the mapping found with the default strategy.

See Mapping Request and Response Body Schemas [page 147] for request body description.

Request Content	Object Mapping Used
No mapping and strategies	The mapping found by the default strategy
Strategies but no mapping	The mapping found by the strategies
A mapping but no strategies	The mapping overloads the default one
A mapping and strategies	The mapping overloads the one found by the strategies

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example: To Change the Data Source by Providing a Mapping (XML)

```
POST /documents/9455/dataproviders/mappings?
originDataproviderIds=DP0&targetDatasourceId=11414
```

In the following request body, there are no data source parameters and all sources are mapped to a target:

```
<mappings>
    <content>
        <mapping>
            <source>
                <id>DS0.DO12</id>
            </source>
            <target>
                <id>DS1.DO12</id>
            </target>
        </mapping>
        <mapping>
            <source>
                <id>DS0.DO2</id>
            </source>
            <target>
                 <id>DS1.DO2</id>
            </target>
        </mapping>
        <mapping>
            <source>
                <id>DS0.DO6</id>
            </source>
            <target>
                <id>DS1.DO39</id>
            </target>
        </mapping>
    </content>
</mappings>
```

Response:

```
<success>
    <message>The resource of type "Document" with identifier "9455" has been
successfully updated.</message>
    <id>9455</id>
</success>
```

Example: To Change the Data Source Where one Source is not Mapped to a Target (XML)

```
POST /documents/9460/dataproviders/mappings?
originDataproviderIds=DP0,DP1&targetDatasourceId=11414
```

In the following request body, there are no data source parameters and the last source (DS0.DO6) is not mapped to a target. The data source object will be removed from the document.

```
<mappings>
    <content>
        <mapping>
            <source>
                <id>DS0.DO12</id>
            </source>
            <target>
                <id>DS1.DO12</id>
            </target>
        </mapping>
        <mapping>
            <source>
                <id>DS0.DO2</id>
            </source>
            <target>
                <id>DS1.DO2</id>
            </target>
        </mapping>
        <mapping>
            <source>
                <id>DS0.DO6</id>
            </source>
            <target/>
        </mapping>
    </content>
</mappings>
```

Example: To Change the Data Source with a Strategy (JSON)

```
POST 16706/dataproviders/mappings?originDataproviderIds=DP0&targetDatasourceId=6610
```

Request body:

```
{
   "mappings":{
      "policy":{
         "strategies":{
            "strategy":[{
```

```
                "@name":"SameId"
            }]
         }
      },
      "content":{
         "mapping":{
            "source":{
               "id":"M[Measures].[Sales Amount]"
            },
            "target":{
               "id":"M[Measures].[Internet Sales Amount]"
            }
         }
      }
   }
}
```

Response:

```
{"success":
    {"message": "The resource of type \"Document\" with identifier \"16706\" has
been successfully updated.",
     "id": "16706"
    }
}
```

Example: To Change the Data Source of a Document with Data Source Parameters (XML)

Data source parameters appear in the response as long as they need to be filled in the request body. When all the parameters have values, then the response is successful.

```
POST /documents/15630/dataproviders/mappings?
targetDatasourceId=6576;ROLE_AA_QRY_VAR_S_OPT
```

The request body is empty. The default mapping is used.

The response contains a parameter:

```
<mappings>
    <parameters>
        <parameter optional="true" type="sapVariable" dpId="DS1">
            <id>1</id>
            <technicalName>ROLE_VAR_20090709102231</technicalName>
            <name>Select a country</name>
            <answer constrained="false" type="Text" keyType="Text">
                <info cardinality="Single" keepLastValues="true">
                    <lov hierarchical="true" nodeSelection="Any"
refreshable="true"/>
                </info>
            </answer>
        </parameter>
    </parameters>
</mappings>
```

Send the following request with the parameter answer:

```
POST /documents/15630/dataproviders/mappings?
targetDatasourceId=6576;ROLE_AA_QRY_VAR_S_OPT
```

Request body:

```
<mappings>
    <parameters>
        <parameter>
            <id>1</id>
            <answer>
                <values>
                    <value id='1'>Argentina</value>
                </values>
            </answer>
        </parameter>
    </parameters>
</mappings>
```

Response:

```
<success>
    <message>The resource of type "Document" with identifier "15630" has been
successfully updated.</message>
    <id>15630</id>
</success>
```

Related Information

Change Source [page 145]
Getting the List of Documents [page 226]
Getting the List of Data Providers [page 555]

8.6.8 Getting the Flow Count of a Data Provider

Usage

Gets the flow count of a data provider.

Request

```
GET /documents/<documentID>/dataproviders/<dataProviderID>/flows/count
```

Response

Response type: `text/plain`

Response body: the number of flows as an integer (from 1 to n).

Example

This example retrieves the flow count for data provider ID DP0 available for the document ID 7738.

```
GET /documents/7738/dataproviders/DP0/flows/count
```

Response

```
1
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Data Providers [page 555]

8.6.9 Getting the Details of a Flow

Usage

Gets the details on the data provider flow specified in XML or CSV format.

Request

```
GET /documents/<documentID>/dataproviders/<dataProviderID>/flows<flowID>
```

Response

Response type: `text/xml` or `text/plain`

Response body: for one data provider, it retrieves the flow details.

- In XML format, the index cells and their values by row.
- In CSV format, values only.

Example: Plain Text

This example retrieves the first flow (`flowId` is `0`) of the data provider `DP0` available for the document ID `7744`.

```
GET /documents/7744/dataproviders/DP0/flows/0
```

Response (CSV format):

```
"Year";"State";"Sales revenue";"Margin"
"2001";"California";"1704210.8";"774893.4"
"2001";"Colorado";"448301.5";"203700.6"
"2001";"DC";"693210.5";"310356.2"
"2001";"Florida";"405985.1";"192479.3"
"2001";"Illinois";"737914.2";"348749.8"
"2002";"California";"2782679.5";"1076528"
"2002";"Colorado";"768389.5";"294482.6"
"2002";"DC";"1215158";"457230.6"
"2003";"California";"2992679";"1121488.5"
...
```

Example: XML Format

This example retrieves the first flow (`flowId` is `0`) of the data provider `DP0` available for the document ID `12575`.

```
GET /documents/12575/dataproviders/DP0/flows/0
```

```
<DATA_PROVIDERS>
    <DATA_PROVIDER>
        <ROW>
            <CELL INDEX="0">2006</CELL>
            <CELL INDEX="1">12</CELL>
            <CELL INDEX="2">Texas </CELL>
            <CELL INDEX="3">Dallas</CELL>
            <CELL INDEX="4">e-Fashion Dallas</CELL>
            <CELL INDEX="5">43302.1</CELL>
            <CELL INDEX="6">222</CELL>
            <CELL INDEX="7">17829.4</CELL>
        </ROW>
        <ROW>
            <CELL INDEX="0">2006</CELL>
            <CELL INDEX="1">12</CELL>
            <CELL INDEX="2">Texas </CELL>
            <CELL INDEX="3">Houston</CELL>
            <CELL INDEX="4">e-Fashion Houston</CELL>
            <CELL INDEX="5">55454.6</CELL>
            <CELL INDEX="6">258</CELL>
            <CELL INDEX="7">24614.6</CELL>
        </ROW>
        <ROW>
            <CELL INDEX="0">2006</CELL>
            <CELL INDEX="1">12</CELL>
            <CELL INDEX="2">Texas </CELL>
            <CELL INDEX="3">Houston</CELL>
            <CELL INDEX="4">e-Fashion Houston Leighton</CELL>
            <CELL INDEX="5">77237.7</CELL>
            <CELL INDEX="6">366</CELL>
            <CELL INDEX="7">34926.7</CELL>
        </ROW>
    </DATA_PROVIDER>
</DATA_PROVIDERS>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Data Providers [page 555]
Getting the Flow Count of a Data Provider [page 583]

8.6.10 Getting the Samples of a Flow

Usage

Returns the data samples of a data provider.

In Web Intelligence, the data samples display in the Query Panel.

> ⚠ Caution
>
> The following method is only supported for Microsoft Excel and free-hand SQL data providers.

Request

```
GET /documents/<documentID>/dataproviders/<dataProviderID>/flows<flowID>/samples
```

Response

Response type: `application/xml` or `application/json`

The response body contains:

- The list of the column names with identifiers and data type (`<column>`)
- The list of column values for each row of the data samples (`<cvalue>`)

Example

```
<samples>
    <columns>
        <column id="0" type="String">Country</column>
        <column id="1" type="String">Region</column>
        <column id="2" type="String">City</column>
        <column id="3" type="Numeric">Revenue</column>
        <column id="4" type="String">Retailer Name</column>
        ...
    </columns>
```

```
    <cvalues>
        <cvalue>
            <column id="0">United States</column>
            <column id="1">California</column>
            <column id="2">Los Angeles</column>
            <column id="3">155000</column>
            <column id="4">Bedford</column>
            ...
        </cvalue>
        <cvalue>
            <column id="0">United States</column>
            <column id="1">California</column>
            <column id="2">Los Angeles</column>
            <column id="3">120150</column>
            <column id="4">Jones</column>
            ...
        </cvalue>
        <cvalue>
            <column id="0">United States</column>
            <column id="1">California</column>
            <column id="2">Los Angeles</column>
            <column id="3">227000</column>
            <column id="4">Smith</column>
            ...
        </cvalue>
        ...
    </cvalues>
</samples>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Data Providers [page 555]
Getting the Flow Count of a Data Provider [page 583]

8.6.11 Getting the Query Plan

Usage

Returns the query plan of the given data provider.

The query plan represents the SQL statements that compose the query.

Request

```
GET /documents/<documentID>/dataproviders/<dataProviderID>/queryplan
```

Response

Response type: `application/xml` or `application/json`

Response body: the details of the query plan, which is made of a series of SQL statements combined using joins and combination operators such as UNION, INTERSECT, and MINUS.

Example

```
GET /documents/9106/dataproviders/DP2/queryplan
```

```
<queryplan custom="false" editable="true">
    <union>
        <fullOuterJoin>
            <statement index="1">SELECT 'FY' || to_char(SALES.invoice_date,'yy'),
                count(distinct SALES.inv_id) FROM SALES GROUP BY 'FY' ||
                to_char(SALES.invoice_date,'yy')</statement>
            <statement index="2">SELECT 'FY' || to_char(SALES.invoice_date,'yy'),
                sum(INVOICE_LINE.nb_guests) FROM SALES, INVOICE_LINE,
SERVICE_LINE,
                SERVICE WHERE (SALES.INV_ID=INVOICE_LINE.INV_ID) AND
                (INVOICE_LINE.SERVICE_ID=SERVICE.SERVICE_ID) AND
                (SERVICE.SL_ID=SERVICE_LINE.SL_ID) AND
                (SERVICE_LINE.service_line ='Accommodation')
                GROUP BY 'FY' || to_char(SALES.invoice_date,'yy')</statement>
        </fullOuterJoin>
        <fullOuterJoin>
            <statement index="3">...</statement>
            <statement index="4">...</statement>
        </fullOuterJoin>
        <intersect>
            <fullOuterJoin>
                <statement index="5">...</statement>
                <statement index="6">...</statement>
            </fullOuterJoin>
            <fullOuterJoin>
                <statement index="7">...</statement>
                <statement index="8">...</statement>
            </fullOuterJoin>
            <minus>
                <fullOuterJoin>
                    <statement index="9">...</statement>
                    <statement index="10">...</statement>
                </fullOuterJoin>
                <fullOuterJoin>
                    <statement index="11">...</statement>
                    <statement index="12">...</statement>
                </fullOuterJoin>
            </minus>
        </intersect>
    </union>
</queryplan>
```

Example

If the data provider has contexts, they must be resolved first.

```
GET /documents/8722/dataproviders/DP5/queryplan
```

Response

```
<error>
    <error_code>WSR 00103</error_code>
    <message>Missing contexts for the data provider "DP5".</message>
</error>
```

```
GET /documents/8722/dataproviders/DP5/parameters?refresh=false
```

The request body contains an answer to the context:

```
<parameters>
    <parameter>
        <id>0</id>
        <answer>
            <values>
                <value id='_VlxrQPzQEeG4Q-3y4CsnNg'>INVOICE_LINE</value>
            </values>
        </answer>
    </parameter>
</parameters>
```

The response shows a parameter of type `prompt`.

This parameter does not need to be answered at this level:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP5">
        <id>1</id>
        <technicalName>pmEnter value(s) for Service</technicalName>
        <name>Enter value(s) for Service</name>
        <answer constrained="false" type="Text">
            <info cardinality="Multiple">
                <lov hierarchical="false" partial="false" refreshable="true"
searchable="true">
                    <id>UNIVERSELOV_DS2.DO27</id>
                    <updated>2014-08-05T12:41:14.000+02:00</updated>
                    <values>
                        <value>Activities</value>
                        <value>Bungalow</value>
                        <value>Car Rent</value>
                        <value>Excursion</value>
                        <value>Fast Food</value>
                        <value>Hotel Room</value>
                        <value>Hotel Suite</value>
                        <value>Poolside Bar</value>
                        <value>Restaurant</value>
                        <value>Sports</value>
                        <value>Travel Reservation</value>
                    </values>
                    <columns mappingId="0">
                        <column id="0" type="String">Service</column>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

```
GET /documents/8722/dataproviders/DP5/queryplan
```

Response:

```
<queryplan custom="false" editable="true">
    <statement index="1">
        SELECT "CITY"."CITY", "SERVICE"."PRICE" FROM "CITY",
        "SERVICE", "INVOICE_LINE", "SALES", "CUSTOMER" WHERE
( "INVOICE_LINE"."SERVICE_ID"
        ="SERVICE"."SERVICE_ID" ) AND
( "SALES"."INV_ID"="INVOICE_LINE"."INV_ID" ) AND
        ( "CUSTOMER"."CUST_ID"="SALES"."CUST_ID" ) AND
( "CITY"."CITY_ID"="CUSTOMER"."CITY_ID" )
        AND "SERVICE"."SERVICE" IN @Prompt('Enter value(s) for
Service','A','Service\Service',
        Multi,Free,Not_Persistent,,User:0)
    </statement>
</queryplan>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Data Providers [page 555]

8.6.12 Updating the Query Plan

Usage

Updates the query plan of the given data provider.

The query plan represents the SQL statements that compose the query.

Request

`PUT /documents/<documentID>/dataproviders/<dataProviderID>/queryplan`

Request body: the details of the query plan.

```
<queryplan custom="false" editable="true">
```

> → Remember
>
> You can only update a query plan with `editable` attribute set to `true`.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

> **Note**
>
> The `custom` attribute is automatically set to `true` once the query plan has been modified.

Restoring the Query Plan

You can revert to the original query plan by passing a request body that contains the `custom` attribute set to `false`.

```
<queryplan custom="false">
```

Example

```
PUT /documents/9178/dataproviders/DP3/queryplan
```

Request body:

```
<queryplan>
    <union>
        <fullOuterJoin>
            <statement index="1">SELECT 'FY' ||
to_char(SALES.invoice_date,'yy'),
                count( distinct SALES.inv_id) FROM SALES GROUP BY 'FY' ||
                to_char(SALES.invoice_date,'yy')</statement>
            <statement index="2">SELECT 'FY' ||
to_char(SALES.invoice_date,'yy'),
                sum(INVOICE_LINE.nb_guests) FROM SALES, INVOICE_LINE,
SERVICE_LINE,
                SERVICE WHERE (SALES.INV_ID=INVOICE_LINE.INV_ID) AND
                (INVOICE_LINE.SERVICE_ID=SERVICE.SERVICE_ID) AND
                (SERVICE.SL_ID=SERVICE_LINE.SL_ID) AND
                (SERVICE_LINE.service_line ='Accommodation') GROUP BY 'FY' ||
                to_char(SALES.invoice_date,'yy')</statement>
        </fullOuterJoin>
        <fullOuterJoin>
            <statement index="3">SELECT 'FY' || to_char(SALES.invoice_date,'yy'),
                count( distinct SALES.inv_id) FROM SALES GROUP BY 'FY' ||
                to_char(SALES.invoice_date,'yy')</statement>
            <statement index="4">SELECT 'FY' ||
to_char(SALES.invoice_date,'yy'),
                sum(INVOICE_LINE.days * INVOICE_LINE.nb_guests * SERVICE.price)
                FROM SALES, INVOICE_LINE, SERVICE WHERE
(SALES.INV_ID=INVOICE_LINE.INV_ID)
                AND (INVOICE_LINE.SERVICE_ID=SERVICE.SERVICE_ID) GROUP BY 'FY'
||
                to_char(SALES.invoice_date,'yy')</statement>
        </fullOuterJoin>
    </union>
```

```
</queryplan>
```

Response:

```
<success>
    <message>The resource of type "QueryPlan" has been successfully updated.</
message>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Data Providers [page 555]

8.6.13 Getting the Query Specification

Usage

Returns the query specification attached to a given data provider.

The query specification describes the parameters and the result objects of the query.

Request

```
GET /documents/<documentID>/dataproviders/<dataProviderID>/specification
```

Response

Response type: `text/xml`

Response body: details of the query specification. See the example.

Example

```
GET /documents/7738/dataproviders/DP0/specification
```

```
<queryspec:QuerySpec ... xmlns:queryspec="http://com.sap.sl.queryspec"
dataProviderId="DP0">
    <queryParameters>
        <duplicatedRowsProperty activated="true" value="true"/>
```

```
        <maxRetrievalTimeInSecondsProperty value="300"/>
        <maxRowsRetrievedProperty value="90000"/>
        <removeEmptyRowsProperty activated="true" value="true"/>
        <allowOtherUserToEditQueryProperty activated="true"
value="true"/>
        <resetContextOnRefreshProperty activated="true" value="true"/>
        <stripQueryProperty/>
   </queryParameters>
   <queriesTree xsi:type="queryspec:QueryOperatorNode"
queryOperator="Union">
        <children xsi:type="queryspec:QueryDataNode">
            <bOQuery name="Query"
identifier="_1y8aENsVEeGswMB7H6m1Qw">
                <resultObjects identifier="DS0.DObc"
name="Year"/>
                <resultObjects identifier="DS0.DOda"
name="State"/>
                <resultObjects identifier="DS0.DOa5"
name="Lines"/>
                <resultObjects identifier="DS0.DO93" name="Sales
revenue"/>
                <conditionPart/>
            </bOQuery>
        </children>
    </queriesTree>
    <propertyBag key="DUPLICATED_ROWS_UNDEFINED_VALUE" value="-1"/>
    <propertyBag key="RESET_CONTEXT_ON_REFRESH_UNDEFINED_VALUE" value="-1"/>
    <propertyBag key="ALLOW_THE_USER_TO_EDIT_QUERY_UNDEFINED_VALUE" value="-1"/>
</queryspec:QuerySpec>
```

> **Note**
>
> In the case of `.unv` universes, the `resultObjects` identifier is a string made of the object identifier retrieved from the universe outline and prefixed by the `dataSourcePrefix` of the current data provider and retrieved from the data provider outline.

Related Information

Getting the List of Documents [page 226]

Getting the List of Data Providers [page 555]

Getting the Details of a Data Provider [page 557]

Getting the Details of a Universe (Web Intelligence) [page 696]

8.6.14 Updating the Query Specification

Usage

Updates the query specification attached to a data provider.

The query specification describes the parameters and the result objects of the query.

Request

```
PUT /documents/<documentID>/dataproviders/<dataProviderID>/specification
```

Request type: `text/xml`

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/7738/dataproviders/DP0/specification
```

Request body in an XML file:

```
<queryspec:QuerySpec xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:queryspec="http://com.sap.sl.queryspec"
    dataProviderId="DP0">
    <queryParameters>
        <duplicatedRowsProperty activated="true" value="true"/>
        <maxRetrievalTimeInSecondsProperty value="300"/>
        <maxRowsRetrievedProperty value="90000"/>
        <removeEmptyRowsProperty activated="true" value="true"/>
        <allowOtherUserToEditQueryProperty activated="true" value="true"/>
        <resetContextOnRefreshProperty activated="true" value="true"/>
        <stripQueryProperty/>
    </queryParameters>
    <queriesTree xsi:type="queryspec:QueryOperatorNode" queryOperator="Union">
        <children xsi:type="queryspec:QueryDataNode">
            <bOQuery name="Query" identifier="_1y8aENsVEeGswMB7H6m1Qw">
                <resultObjects identifier="DS0.DObc" name="Year"/>
                <resultObjects identifier="DS0.DOda" name="State"/>
                <resultObjects identifier="DS0.DOa5" name="Lines"/>
                <resultObjects identifier="DS0.DO93" name="Sales revenue"/>
                <conditionPart/>
            </bOQuery>
        </children>
    </queriesTree>
    <propertyBag key="DUPLICATED_ROWS_UNDEFINED_VALUE" value="-1"/>
    <propertyBag key="RESET_CONTEXT_ON_REFRESH_UNDEFINED_VALUE" value="-1"/>
    <propertyBag key="ALLOW_THE_USER_TO_EDIT_QUERY_UNDEFINED_VALUE" value="-1"/>
</queryspec:QuerySpec>
```

> **Note**
>
> In the case of `.unv` universes, the `resultObjects` identifier is a string made of the object identifier retrieved from the universe outline and prefixed by the `dataSourcePrefix` of the current data provider and retrieved from the data provider outline.

Response:

```
<success>
    <message>The resource of type "Data provider" with identifier "DP0" has been
successfully updated.</message>
    <id>DP0</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Data Providers [page 555]
Getting the Details of a Data Provider [page 557]
Getting the Details of a Universe (Web Intelligence) [page 696]

8.6.15 Getting the Definition of an Object

Usage

Returns an expression, a variable, a link or a refcell.

Request

```
GET /documents/<documentId>/dataobjects/<dataobjectId>
```

Response

Response type: `application/xml` or `application/json`

Example (XML)

```
GET /documents/1234/dataobjects/DP0.DO84
```

Response:

```
HTTP code: 200
```

```
<expression dataType="String" qualification="Dimension" customSort="Defined"
allowUserValues="true">
    <id>DP0.DO84</id>
    <name>Country</name>
    <description>/All Geographical/Country Object on Country Table, can be used
for multi Dps Test with a Links between Country Id Objects</description>
    <dataSourceObjectId>DS0.DO84</dataSourceObjectId>
    <formulaLanguageId>[Country]</formulaLanguageId>
    <dataProviderId>DP0</dataProviderId>
    <dataProviderName>Query 1</dataProviderName>
    <dataSourceId>6431</dataSourceId>
    <dataSourceName>Beach oracle(Beach_oracle)</dataSourceName>
</expression>
```

8.6.16 Getting the List of Values of a Data Object

Usage

Returns the list of values of a data object of a Web Intelligence document.

A data object is a data provider expression, a variable, or a linked dimension.

Request

`GET /documents/<documentID>/dataobjects/<dataObjectID>/lov?formattedValues=<formattedValues>`

Where:

- `<formattedValues>` is an optional, Boolean parameter. Default value is `false`. If set to `true`, the DateTime and Numeric values are formatted according to the preferred viewing locale set through `X-SAP-PVL` in the request.

Response

Response type: `application/xml` or `application/json`

Example: XML

`GET /documents/6071/dataobjects/DP0.DO84/lov`

Response:

```
<lov mandatorySearch="false" searchable="false" refreshable="false"
partial="false" hierarchical="false">
```

```
    <values>
        <value>Germany</value>
        <value>Japan</value>
        <value>Madagascar</value>
        <value>Middle East</value>
        <value>Nepal</value>
        <value>South Africa</value>
        <value>US</value>
    </values>
    <columns mappingId="0">
        <column type="String" id="0">Country</column>
    </columns>
</lov>
```

Example: JSON

`GET /documents/5152/dataobjects/DP0.DO1fc/lov`

Response:

```
{"lov":{
    "@mandatorySearch":"false",
    "@searchable":"true",
    "@refreshable":"false",
    "@partial":"false",
    "@hierarchical":"false",
    "values":{
        "value":[
            "2 Pocket shirts",
            "Belts,bags,wallets",
            "Bermudas",
            "Boatwear",
            "Cardigan",
            "Casual dresses",
            "Day wear",
            "Dry wear",
            "Evening wear",
            "Fancy fabric",
            "Full length",
            "Hair accessories",
            "Hats,gloves,scarves",
            "Jackets",
            "Jeans",
            "Jewelry",
            "Long lounge pants",
            "Long sleeve",
            "Lounge wear",
            "Mini city",
            "Night wear",
            "Outdoor",
            "Pants",
            "Party pants",
            "Samples",
            "Shirts",
            "Short sleeve",
            "Skirts",
            "Soft fabric",
            "Sweater dresses",
            "Sweats",
            "T-Shirts",
            "Turtleneck",
            "Wet wear"
        ]
```

```
    },
    "columns":{
        "@mappingId":"0",
        "column":[{
            "@type":"String",
            "@id":"0",
            "$":"Category"}]
        }
    }
}
```

Related Information

Getting the List of Documents [page 226]
Getting the Definition of an Input Control [page 533]

8.6.17 Getting the List of Values Under a Specific Data Object Value

Usage

Returns the list of values under a specified value of a data object in a hierarchy.

A data object is a data provider expression, a variable, or a linked dimension.

Request

```
PUT /documents/<documentID>/dataobjects/<dataObjectID>/lov?
formattedValues=<formattedValues>
```

Where:

- `<formattedValues>` is an optional, Boolean parameter. Default value is `false`. If set to `true`, the DateTime and Numeric values are formatted according to the preferred viewing locale set through `X-SAP-PVL` in the request.

Request body: a specific value of the list of values of a data object.

> ⚠ Caution
>
> Make sure the value you specify in the request body is correct. The Web Service SDK does not make any checking of the value. Incorrect values are ignored and replaced with the root values.

Response

Response type: `application/xml` or `application/json`

Example

`PUT /documents/6070/dataobjects/DP0.DO9d/lov`

Request body:

```
<lov hierarchical="true" partial="false" refreshable="false">
    <query>
        <path>
            <value id="\[Customer\].\[Customer Geography\].\[State-Province\|
State-Province\].&\[NSW\]&\[AU\]" final="false">Queensland</value>
        </path>
    </query>
</lov>
```

Response:

```
<lov hierarchical="true" partial="false" refreshable="false">
    <values>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\[Alexandria\]&\
[NSW\]" final="false">Alexandria</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\[Coffs
Harbour\]&\[NSW\]" final="false">Coffs Harbour</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\
[Darlinghurst\]&\[NSW\]" final="false">Darlinghurst</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\[Goulburn\]&\
[NSW\]" final="false">Goulburn</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\[Lane Cove\]&\
[NSW\]" final="false">Lane Cove</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\[Lavender
Bay\]&\[NSW\]" final="false">Lavender Bay</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\[Malabar\]&\
[NSW\]" final="false">Malabar</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\[Matraville\]&\
[NSW\]" final="false">Matraville</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\[Milsons
Point\]&\[NSW\]" final="false">Milsons Point</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\[Newcastle\]&\
[NSW\]" final="false">Newcastle</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\[North Ryde\]&\
[NSW\]" final="false">North Ryde</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\[North
Sydney\]&\[NSW\]" final="false">North Sydney</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\[Port
Macquarie\]&\[NSW\]" final="false">Port Macquarie</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\[Rhodes\]&\
[NSW\]" final="false">Rhodes</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\
[Silverwater\]&\[NSW\]" final="false">Silverwater</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\[Springwood\]&\
[NSW\]" final="false">Springwood</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\[St.
Leonards\]&\[NSW\]" final="false">St. Leonards</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\[Sydney\]&\
[NSW\]" final="false">Sydney</value>
        <value id="\[Customer\].\[Customer Geography\].\[City\].&\[Wollongong\]&\
[NSW\]" final="false">Wollongong</value>
```

```
    </values>
</lov>
```

Related Information

Getting the List of Documents [page 226]

8.7 Managing Personal Data Providers

Data providers are data sources used to build queries in Web Intelligence documents.

You can build documents from Microsoft Excel 2003 or Microsoft Excel 2007 spreadsheets. Before doing so, you need to upload the file to the CMS repository.

For more information about data providers, see Managing Data Providers [page 554].

Uploading a Microsoft Excel File to the CMS Repository [page 600]

Getting the List of Microsoft Excel Files [page 602]

Getting the Details of a Microsoft Excel File [page 603]

Updating a Microsoft Excel File to the CMS Repository [page 605]

Deleting a Microsoft Excel File [page 606]

8.7.1 Uploading a Microsoft Excel File to the CMS Repository

Usage

Uploads and stores a Microsoft Excel file to the CMS repository.

Request

`POST /spreadsheets`

Request type: `multipart/form-data`

The request body is a multipart body made of the following parts:

- The file details, which can be XML or JSON. Only `name`, `folderId` elements are mandatory. `description` is optional. The other spreadsheet elements are ignored.

- The file itself as a binary stream. `Content-Type` is either `application/vnd.ms-excel` for Microsoft Excel 2003, or `application/vnd.openxmlformats-officedocument.spreadsheetml.sheet` for Microsoft Excel 2007.

> **Note**
>
> Make sure the request body has the necessary empty lines.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

XML request body for a Microsoft Excel 2007 file:

```
--10f3211b-66e8-4b14-93ec-b11ec5c19a43-41d3c84c-d157-4377-aba9-41353b9a4dfe
Content-Disposition: form-data; name="attachmentInfos"
Content-Type: application/xml
    // mandatory carriage return
<spreadsheet>
    <name>myfile.xlsx</name>
    <folderId>6206</folderId>
</spreadsheet>
    // mandatory carriage return
--10f3211b-66e8-4b14-93ec-b11ec5c19a43-41d3c84c-d157-4377-aba9-41353b9a4dfe
Content-Disposition: form-data; name="attachmentContent"
Content-Type: application/vnd.openxmlformats-officedocument.spreadsheetml.sheet
    // mandatory empty line
... Excel 2007 document as a binary stream...
    // mandatory carriage return
--10f3211b-66e8-4b14-93ec-b11ec5c19a43-41d3c84c-d157-4377-aba9-41353b9a4dfe--
```

XML response:

```
<success>
    <message>The resource of type "Spreadsheet" with identifier "7466" has been
successfully created.</message>
    <id>7466</id>
</success>
```

JSON request body for a Microsoft Excel 2003 file:

```
--10f3211b-66e8-4b14-93ec-b11ec5c19a43-41d3c84c-d157-4377-aba9-41353b9a4dfe
Content-Disposition: form-data; name="attachmentInfos"
Content-Type: application/json
    // mandatory carriage return
{"spreadsheet":{"name":"myfile.xls","folderId":6206}}
    // mandatory carriage return
--10f3211b-66e8-4b14-93ec-b11ec5c19a43-41d3c84c-d157-4377-aba9-41353b9a4dfe
Content-Disposition: form-data; name="attachmentContent"
Content-Type: application/vnd.ms-excel
```

```
    // mandatory empty line
... Excel 2007 document as a binary stream...
    // mandatory carriage return
--10f3211b-66e8-4b14-93ec-b11ec5c19a43-41d3c84c-d157-4377-aba9-41353b9a4dfe--
```

JSON response:

```
{"success":
    {"message":"The resource of type \"Spreadsheet\" with identifier \"7492\"
has been successfully created.",
     "id":7492
    }
}
```

8.7.2 Getting the List of Microsoft Excel Files

Usage

Returns the Microsoft Excel files stored in the CMS repository.

Request

`GET /spreadsheets?offset=<offset>&limit=<limit>`

Where:

- `<offset>` indicates the position in the list, from which documents are returned. It must be greater than or equal to 0. The default value is 0. This parameter is optional.
- `<limit>` indicates the number of documents in the list. Its range is [0, 50]. The default value is 10. This parameter is optional.

Response

Response type: `application/xml` or `application/json`

Response body: the list of files. Each `<spreadsheet>` element has the following children:

- `<id>`: identifier of the file on the CMS repository
- `<cuid>`: CUID of the file on the CMS repository
- `<name>`: name of the file
- `<description>`: description of the file
- `<folderId>`: identifier of the folder of the CMS repository that contains the file

Example

XML response:

```
<spreadsheets>
    <spreadsheet>
        <id>6535</id>
        <cuid>FljHNFCAVAMAzgUAAFD5eRsAAFBWoELQ</cuid>
        <name>MySheet</name>
        <description>one of my spreadsheets</description>
        <folderId>6532</folderId>
    </spreadsheet>
    <spreadsheet>
        <id>6641</id>
        <cuid>AbbqtvKKLoBFjhvhSqb5S7w</cuid>
        <name>MySheet2</name>
        <folderId>6206</folderId>
    </spreadsheet>
    <spreadsheet>
        <id>6541</id>
        <cuid>FjdAO1D.igIAzgUAAFCJCBoAAFBWoELQ</cuid>
        <name>MySheet3</name>
        <description>one of my spreadsheets</description>
        <folderId>6539</folderId>
    </spreadsheet>
</spreadsheets>
```

JSON response:

```
{"spreadsheets":
    {"spreadsheet":
        [{"id":6535,
          "cuid":"FljHNFCAVAMAzgUAAFD5eRsAAFBWoELQ",
          "name":"MySheet",
          "description":"one of my spreadsheets",
          "folderId":6532},
         {"id":6641,"cuid":"AbbqtvKKLoBFjhvhSqb5S7w",
          "name":"MySheet2",
          "folderId":6206},
         {"id":6541,
          "cuid":"FjdAO1D.igIAzgUAAFCJCBoAAFBWoELQ",
          "name":"MySheet3",
          "description":"one of my spreadsheets",
          "folderId":6539}]
    }
}
```

8.7.3 Getting the Details of a Microsoft Excel File

Usage

Returns the details of a Microsoft Excel file stored on the CMS repository.

Request

```
GET /spreadsheets/<spreadsheetID>
```

Response

Response type: `application/xml` or `application/json`

Response body: the details of the file with the following information:

- `<id>`: file identifier
- `<cuid>`: CUID of the file
- `<name>`: name of the file
- `<folderId>`: identifier of the folder of the CMS repository that contains the file
- `<path>`: path to the file in the CMS repository directory
- `<updated>`: date of last update
- `<createdBy>`: name of spreadsheet creator
- `<file>`: name of the file with an extension
- `<size>`: size of the file in bytes
- `<mimeType>`: mime type (`application/vnd.ms-excel` for Microsoft Excel 2003 or `application/vnd.openxmlformats-officedocument.spreadsheetml.sheet` for Microsoft Excel 2007)
- `<sheet>`: name of the sheet contained in the file
- `<range>`: name of a named cell range in the file

Example

XML response:

```
<spreadsheet>
    <id>7168</id>
    <cuid>AQJqfTLbR5RNoYhzV9DnCMA</cuid>
    <name>myfile</name>
    <folderId>6206</folderId>
    <path>Public Folders/Documents/</path>
    <updated>2014-07-17T14:27:44.367+02:00</updated>
    <createdBy>Administrator</createdBy>
    <file>myfile.xlsx</file>
    <size>22444</size>
    <mimeType>application/vnd.ms-excel</mimeType>
    <sheets>
        <sheet>sheet1</sheet>
        <sheet>sheet2</sheet>
        <sheet>sheet3</sheet>
        <sheet>sheet4</sheet>
    </sheets>
    <namedRanges>
        <range>items</range>
        <range>region</range>
        <range>region_item_units_cost</range>
```

```
        <range>sum_cost</range>
        <range>sum_units</range>
    </namedRanges>
</spreadsheet>
```

JSON response:


```
{"spreadsheet":
    {"id":7168,
     "cuid":"AQJqfTLbR5RNoYhzV9DnCMA",
     "name":"myfile",
     "folderId":6206,
     "path":"Public Folders\/Documents\/",
     "updated":"2014-07-17T10:34:19.763+02:00",
     "createdBy":"Administrator",
     "file":"myfile.xlsx",
     "size":22444,
     "mimeType":"application\/vnd.ms-excel",
     "sheets":
         [{"sheet":"sheet1"},
          {"sheet":"sheet2"},
          {"sheet":"sheet3"},
          {"sheet":"sheet4"}],
     "namedRanges":
         [{"range":"items"},
          {"range":"region"},
          {"range":"region_item_units_cost"},
          {"range":"sum_costs"},
          {"range":"sum_units"}]
    }
}
```


Related Information

Getting the List of Microsoft Excel Files [page 602]

8.7.4 Updating a Microsoft Excel File to the CMS Repository

Usage

Re-uploads a Microsoft Excel file stored on the CMS repository with a different content.

> → Remember
>
> The current file and the one to upload must have the same file structure (columns number, names, and order).

Request

`PUT /spreadsheets/<spreadsheetID>`

The request type is:

- `application/vnd.ms-excel` for Microsoft Excel 2003
- `application/vnd.openxmlformats-officedocument.spreadsheetml.sheet` for Microsoft Excel 2007

The request body only contains the Microsoft Excel file as a binary stream. You do not need to send the file details as attachment again.

> **Note**
>
> Make sure the request body has the necessary empty lines.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

XML request body for a Microsoft Excel 2007 file:

```
... Excel 2007 document as a binary stream...
```

XML response:

```
<success>
    <message>The resource of type "Spreadsheet" with identifier "7466" has been
successfully updated.</message>
    <id>7466</id>
</success>
```

8.7.5 Deleting a Microsoft Excel File

Usage

Deletes a Microsoft Excel file stored on the CMS repository.

Request

```
DELETE /spreadsheets/<spreadsheetID>
```

Response

Response type: `application/xml` or `application/json`

Response body: The response is a message stating the success or failure of the request.

Example

```
DELETE /spreadsheets/8995
```

```
<success>
    <message>The resource of type "Spreadsheet" with identifier "8995" has been
successfully removed.</message>
    <id>8995</id>
</success>
```

Related Information

Getting the List of Microsoft Excel Files [page 602]

8.8 Managing SAP BW Connections and BEx Queries

APIs are provided to perform tasks on SAP BW connections and BEx queries.

Getting the List of SAP BW Connections [page 607]

Getting the Details of an SAP BW Connection [page 609]

Browsing the Details of an SAP BW connection [page 610]

Getting the Outline of a BEx Query [page 612]

Getting the Capabilities of a BEx Query [page 614]

8.8.1 Getting the List of SAP BW Connections

Usage

Returns the list of available SAP BW connections. You can only see the connections that you have authorization to see.

Request

```
GET /bwconnections?offset=<offset>&limit=<limit>
```

Where:

- `<offset>` is the offset from the beginning of the list. Default value is 0.
- `<limit>` is the maximum number of connections to return. Default value is 10.

Response

Response type: `application/xml` or `application/json`

Response body: the list of BW connections that you have the authorization to see/access.

Example

```
<bwconnections>
    <bwconnection type="Cube">
        <id>7052</id>
        <cuid>AdDDU67.DyxBkOgpzjDJn30</cuid>
        <name>Adventure Works MSAS2005</name>
        <folderId>7131</folderId>
    </bwconnection>
    <bwconnection type="Cube">
        <id>11540</id>
        <cuid>AbUAJD7zVpZFgy2jQQNMyI8</cuid>
        <name>AdventureWorks</name>
        <folderId>11484</folderId>
    </bwconnection>
    <bwconnection type="Cube">
        <id>11537</id>
        <cuid>AROnvrBXn1tOpsXj_jPPtYo</cuid>
        <name>ADW</name>
        <folderId>11484</folderId>
    </bwconnection>
    ...
    <bwconnection type="Query">
        <id>7268</id>
        <cuid>AUeWmURZfzVKjnpwfTWVxvM</cuid>
        <name>BICS_Query</name>
        <folderId>4066</folderId>
    </bwconnection>
    <bwconnection type="System">
        <id>7039</id>
        <cuid>AY2UQVyb3WRLioC8GBg5Vi0</cuid>
        <name>BICS_Server</name>
        <folderId>4066</folderId>
    </bwconnection>
</bwconnections>
```

8.8.2 Getting the Details of an SAP BW Connection

Usage

Returns the details of an SAP BW connection.

Request

```
GET /bwconnections/<bwConnectionID>
```

Response

Response type: `application/xml` or `application/json`

Response body: the details of an SAP BW connection that you have the authorization to access.

`<path>` is the connection path in the CMS repository.

Example: SAP BW Connection of Type Query

```
GET /bwconnections/11489
```

```
<bwconnection type="Query">
    <id>11489</id>
    <cuid>AXRu2fNiQphAtF4O4lJ4OKg</cuid>
    <name>bex_simple</name>
    <folderId>11484</folderId>
    <path>Application Folder/Root Folder/Connections/</path>
    <bwnodes>
        <bwnode type="Query">
            <id>11489</id>
            <name>A simple Bex Query</name>
            <technicalName>QRY_SIMPLE</technicalName>
            <path>/QRY_SIMPLE</path>
        </bwnode>
    </bwnodes>
</bwconnection>
```

Example: Incomplete SAP BW Connection of Type System or Cube

```
GET /bwconnections/11990
```

```
<bwconnection type="System">
    <id>11990</id>
```

```
    <cuid>Aaj0N_I.bSJElyDKk08sxTU</cuid>
    <name>raylight_BOF</name>
    <folderId>11484</folderId>
    <path>Application Folder/Root Folder/Connections/</path>
    <bwnodes>
        <bwnode type="Favorites">
            <name>Favorites</name>
            <technicalName>SystemFavoritesTopLevel</technicalName>
            <path>/SystemFavoritesTopLevel</path>
        </bwnode>
        <bwnode type="InfoArea">
            <name>InfoArea</name>
            <technicalName>SystemInfoareaTopLevel</technicalName>
            <path>/SystemInfoareaTopLevel</path>
        </bwnode>
    </bwnodes>
</bwconnection>
```

> **Note**
>
> Complete the connection by searching for a BW node of type Query in the system or cube.

Related Information

Getting the List of SAP BW Connections [page 607]

8.8.3 Browsing the Details of an SAP BW connection

Usage

Browses the details of an incomplete SAP BW connection and helps you choose a BEx query which will complete the connection.

The "by path" and "by search pattern" browsing methods are available.

> **Note**
>
> Only MDX compliant BEx queries are usable in Web Intelligence workflows.

Request

`PUT /bwconnections/<bwConnectionID>`

Request body (optional):

- By path:

```
<bwnode>
```

```
    <path>[path/to/the/required/node]</path>
</bwnode>
```

- By search pattern:

```
<bwnode>
    <pattern>[pattern to be used for the search]</pattern>
</bwnode>
```

Response

Response type: `application/xml` or `application/json`

Response body: the list of BW connections that you have the authorization to access.

> **Note**
>
> A BEx query identifier is defined by the concatenation of the BW connection identifier and the technical name of the BEx query, separated by a semi-colon (`;`).

Example: 1: To get the list of BEx queries inside a BW connection, browsing by path

`PUT /bwconnections/11990`

Request body in an XML file:

```
<bwnode>
    <path>/SystemInfoareaTopLevel/BUSINESSOBJECTS_QA/Z_BOBJ/TEST_DATE</path>
</bwnode>
```

Response:

```
<bwconnection type="System">
    <id>11990</id>
    <cuid>Aaj0N_I.bSJElyDKk08sxTU</cuid>
    <name>raylight_BOF</name>
    <folderId>11484</folderId>
    <bwnodes>
        <bwnode type="Query">
            <id>11990;TEST_DATE</id>
            <name>test_data</name>
            <technicalName>TEST_DATE</technicalName>
            <mdxCompliant>true</mdxCompliant>
            <path>/SystemInfoareaTopLevel/BUSINESSOBJECTS_QA/Z_BOBJ/TEST_DATE</
path>
        </bwnode>
    </bwnodes>
</bwconnection>
```

Example: To get the list of BEx queries inside a BW connection matching a pattern

```
PUT /bwconnections/11990
```

Request body in an XML file:

```
<bwnode>
    <pattern>*BOBJ*</pattern>
</bwnode>
```

Response:

```
<bwconnection type="System">
    <id>11990</id>
    <cuid>Aaj0N_I.bSJElyDKk08sxTU</cuid>
    <name>raylight_BOF</name>
    <folderId>11484</folderId>
    <bwnodes>
        <bwnode type="Query">
            <id>11990;Z_BOBJ;BOBJ_TEST_ROLE</id>
            <name>BOBJ_TEST_ROLE</name>
            <technicalName>BOBJ_TEST_ROLE</technicalName>
            <mdxCompliant>false</mdxCompliant>
        </bwnode>
        <bwnode type="Query">
            <id>11990;Z_BOBJ;IE_ZBOBJ_COUNTRYVAR</id>
            <name>IE_ZBOBJ_COUNTRYVAR</name>
            <technicalName>IE_ZBOBJ_COUNTRYVAR</technicalName>
            <mdxCompliant>true</mdxCompliant>
        </bwnode>
        <bwnode type="Query">
            <id>11990;T_BOBJ;ZCR_BOBJ_ALLF_MHIER_FCHAR_1</id>
            <name>ZCR BOBJ all_fields mult_hier free_chars</name>
            <technicalName>ZCR_BOBJ_ALLF_MHIER_FCHAR_1</technicalName>
            <mdxCompliant>false</mdxCompliant>
        </bwnode>
        <bwnode type="Query">
            <id>11990;Z_BOBJ;Z_BOBJ_HB_SIMPLE</id>
            <name>Z_BOBJ_HB_simple</name>
            <technicalName>Z_BOBJ_HB_SIMPLE</technicalName>
            <mdxCompliant>true</mdxCompliant>
        </bwnode>
    </bwnodes>
</bwconnection>
```

8.8.4 Getting the Outline of a BEx Query

Usage

Returns the details of a BEx query.

Request

`PUT /bwconnections/<bwConnectionID>/outline`

Request body (optional):

```
<bwnode>
    <id>
```

Where:

- `<id>` is the BEx query identifier

Response

Response type: `application/xml` or `application/json`

Response body: the outline of a BEx query that you have the authorization to see/access.

Example

`PUT /bwconnections/11990/outline`

Request body in an XML file:

```
<bwnode>
    <id>11990;ROLE_ST_BEX5</id>
</bwnode>
```

Response:

```
<outline>
    <item type="Dimension">
    <name>Country</name>
    <description>Country</description>
    <item type="Hierarchy">
        <name>Country</name>
        <description>Country</description>
        <id>HZ_COUNTRY</id>
    </item>
    <item type="Hierarchy">
        <name>Country Hierarchy 01</name>
        <description>Country Hierarchy 01</description>
        <item type="Level">
            <name>Level 01</name>
            <description>Level 01</description>
            <id>LCOUNTRY_HIERARCHY_01|Z_COUNTRY.#1</id>
        </item>
        <item type="Level">
            <name>Level 02</name>
            <description>Level 02</description>
            <id>LCOUNTRY_HIERARCHY_01|Z_COUNTRY.#2</id>
        </item>
        <item type="Level">
```

```
            <name>Level 03</name>
            <description>Level 03</description>
            <id>LCOUNTRY_HIERARCHY_01|Z_COUNTRY.#3</id>
        </item>
        <id>HCOUNTRY_HIERARCHY_01|Z_COUNTRY</id>
    </item>
    . . .
    <item type="Measure">
        <name>Voyager Training 01 - Restricted KF</name>
        <description>Voyager Training 01 - Restricted KF</description>
        <item type="Attribute">
            <name>Voyager Training 01 - Restricted KF Currency</name>
            <id>MD4NUM119ATNWC7USQ4XYSA0QY.Currency</id>
        </item>
        <item type="Attribute">
            <name>Voyager Training 01 - Restricted KF Formatted Value</name>
            <id>MD4NUM119ATNWC7USQ4XYSA0QY.FormattedValue</id>
        </item>
        <id>MD4NUM119ATNWC7USQ4XYSA0QY</id>
    </item>
</outline>
```

Related Information

Getting the List of SAP BW Connections [page 607]

8.8.5 Getting the Capabilities of a BEx Query

Usage

Returns the capabilities of a BEx query.

Request

`PUT /bwconnections/<bwConnectionID>/capabilities`

Request body (optional):

```
<bwnode>
    <id>
```

Where:

- `<id>` is the BEx query identifier

Response

Response type: `application/xml` or `application/json`

Response body: the capabilities of a BEx query that you have the authorization to see/access.

Example

`PUT /bwconnections/11990/capabilities`

Request body in an XML file:

```
<bwnode>
    <id>11990;Z_BOBJ;ROLE_ST_BEX5</id>
</bwnode>
```

Response:

```
<datasource:QueryCapability xmlns:datasource="http://com.sap.sl.datasource">
    <generalCapability customQueryScriptSupported="false"
showHideScopeSupported="false"/>
    <dataProcessingCapability removeEmptyRowsAvailable="true"
maxRetrievalTimeAvailable="false"
        queryStrippingAvailable="true"/>
    <filterCapability resultHierarchyInFilterSupported="false"
constantComparisonSupported="true"
        hierarchyConstantOperandSupported="true">
        <supportedComparisonOperators>equal</supportedComparisonOperators>
        <supportedComparisonOperators>notEqual</supportedComparisonOperators>
        ...
        <supportedValueBasedHierarchyComparisonOperators>equal</
supportedValueBasedHierarchyComparisonOperators>
        <supportedValueBasedHierarchyComparisonOperators>inList</
supportedValueBasedHierarchyComparisonOperators>
        <supportedLevelBasedHierarchyComparisonOperators>equal</
supportedLevelBasedHierarchyComparisonOperators>
        <supportedLevelBasedHierarchyComparisonOperators>inList</
supportedLevelBasedHierarchyComparisonOperators>
        <supportedLogicalOperators>and</supportedLogicalOperators>
        <supportedObjects>attribute</supportedObjects>
        <supportedObjects>dimension</supportedObjects>
        <supportedObjects>hierarchy</supportedObjects>
        <supportedObjects>level</supportedObjects>
    </filterCapability>
    <resultObjectCapability useAttributeSeparatelyAvailable="true"
buildLogicalSetWithMetadataFunctionsAvailable="true"
        buildLogicalSetWithMemberFunctionsAvailable="true"
memberSelectionPromptSupported="true"
        supportsMemberSelectionCompletion="true"/>
</datasource:QueryCapability>
```

Related Information

Getting the List of SAP BW Connections [page 607]

8.9 Managing Connections for Free-Hand SQL Data Providers

Data providers are data sources used to build queries in Web Intelligence documents.

You can build documents directly from SQL scripts on the top of relational secured database connections without using universes. This feature allows you to support rich and complex SQL queries using advanced specific-database functions.

For more information about data providers, see Managing Data Providers [page 554].

Getting the List of Connections [page 616]

Getting the Details of a Connection [page 618]

8.9.1 Getting the List of Connections

Usage

Returns the list of connections stored in the CMS repository that the end-user has access to.

Request

```
GET /connections?type=<type>&offset=<offset>&limit=<limit>
```

Where:

- `<type>` indicates the connection type to be retrieved. Values are `Relational`, `FlattenedOlap`, `Olap` or `DataFederator`. This parameter is optional. If not specified, all types of connections are retrieved.
- `<offset>` indicates the position in the list, from which connections are returned. It must be greater than or equal to 0. The default value is 0. This parameter is optional.
- `<limit>` indicates the number of connections that you can list on one page. Its range is [1, 50]. This parameter is optional. The default value is 10.

Response

Response type: `application/xml` or `application/json`

Example: Request Without Parameters

```
GET /connections
```

XML response:

```
<connections>
    <connection type="Olap">
        <id>6100</id>
        <cuid>AbpsL1MVilFPoM.RfGB02m0</cuid>
        <name>AAQUEY_HIERNODE_01</name>
        <folderId>6069</folderId>
    </connection>
    <connection type="Relational">
        <id>5993</id>
        <cuid>AbL85pdeN7NOqNjd1j0btlA</cuid>
        <name>BOF_ERP_en_UK_Save_Lang_IDT</name>
        <folderId>5971</folderId>
    </connection>
    <connection type="FlattenedOlap">
        <id>6118</id>
        <cuid>ARgmu_R0AQhDvJFmrHnP5F0</cuid>
        <name>BW_VAR_FORMULA</name>
        <folderId>6069</folderId>
    </connection>
    <connection type="DataFederator">
        <id>6196</id>
        <cuid>AYATF1eY_8RPt0LWH4Vcfug</cuid>
        <name>DF data source BW</name>
        <folderId>548</folderId>
    </connection>
    ...
</connections>
```

Example: Request With Parameters

```
GET /connections?limit=3&offset=2&type=FlattenedOlap
```

JSON response:

```
{"connections":
    {"connection":
        [{"@type":"FlattenedOlap",
          "id":6117,
          "cuid":"AZHZlbdSX85LmE3U7dsC120",
          "name":"BW_VAR_KD_DEFVAL",
          "folderId":6069},
         {"@type":"FlattenedOlap",
          "id":6112,
          "cuid":"Adr6Epit67FPu_0JebFfgqY",
          "name":"BW_VAR_RANGE",
          "folderId":6069},
         {"@type":"FlattenedOlap",
          "id":6110,
          "cuid":"AQ5UAwvV1ptJhuNBzYmdXTw",
          "name":"BW_VAR_TEXT",
          "folderId":6069}
        ]
    }
}
```

8.9.2 Getting the Details of a Connection

Usage

Returns the details of a connection stored in the CMS repository.

Request

```
GET /connections/<connectionID>
```

Response

Response type: `application/xml` or `application/json`

Example

```
GET /connections/6224
```

XML response:

```
<connection type="Relational">
    <id>6224</id>
    <cuid>AZKgD3.WRiJAhYfy7vjCZ_A</cuid>
    <name>beach</name>
    <folderId>6069</folderId>
    <path>Connections/MyConnections</path>
    <updated>2014-08-06T14:11:02.000+02:00</updated>
    <createdBy>Administrator</createdBy>
    <database>Oracle 10</database>
    <networkLayer>Oracle OCI</networkLayer>
</connection>
```

JSON response:

```
{"connection":
    {"@type":"Relational",
     "id":6224,
     "cuid":"AZKgD3.WRiJAhYfy7vjCZ_A",
     "name":"beach",
     "folderId":6069,
     "path":"Connections\/MyConnections",
     "updated":"2014-08-06T14:11:02.000+02:00",
     "createdBy":"Administrator",
     "database":"Oracle 10",
     "networkLayer":"Oracle OCI"}
}
```

Related Information

Getting the List of Connections [page 616]

8.10 Refreshing Documents

Below are the refresh operations you can do on a Web Intelligence document:

- Refreshing a document without contexts and prompts
- Identifying contexts and prompts of a document
- Fill in contexts and prompts with values

Supported prompts are the following:

- Prompts that accept either string, numeric or date values
- Prompts that accept one value or multiple values
- Optional and non optional prompts

Getting the Refresh Parameters of a Document [page 619]

Refreshing a Document [page 626]

Cancelling the Refresh of a Document [page 647]

Getting the Details of a Parameter [page 649]

Working with Variants [page 653]

8.10.1 Getting the Refresh Parameters of a Document

Usage

Returns the parameters to be filled before running a document refresh.

Request

```
GET /documents/<documentID>/parameters?<optional_parameters>
```

Where:

Optional Parameters

Parameter	Description
`formattedValues`	Optional, Boolean parameter. Default value is `false`. If set to `true`, the Date-Time and Numeric values are formatted according to the preferred viewing locale set through `X-SAP-PVL` in the request.
`lovInfo`	Optional, Boolean parameter. Default value is `true`. If set to `false`, the lists of values are not computed, nor displayed.
`summary`	Optional, Boolean parameter. Default value is `false`. If set to true, a summary of the previous values will be returned.

Response

Response type: `application/xml` or `application/json`

Response body: the response provides the parameters with their expected answers, previous values if applicable, otherwise default values. See Parameter Response Body Schemas [page 133]

Example: No Parameter

Response:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<parameters/>
```

Example: Context

The parameter of type `context` provides two possible context values.

Response:

```
<parameters>
    <parameter optional="false" type="context" dpId="DP0">
        <id>0</id>
        <technicalName>0</technicalName>
        <name>Select a context</name>
        <answer constrained="true" type="Text">
            <info cardinality="Multiple">
                <lov hierarchical="false" partial="false" refreshable="true"
searchable="false"/>
                    <values>
                        <value id="2">Reservations</value> <!-- possible values
-->
                        <value id="1">Sales</value>
                    </values>
                </lov>
                <previous>
                    <value id="2">Reservations</value> <!-- previous value -->
```

```
                </previous>
            </info>
            <values>
                <value id="2">Reservations</value> <!-- previous value -->
            </values>
        </answer>
    </parameter>
</parameters>
```

Example: DateTime Prompt

The parameter of type `prompt` accepts only one answer value (cardinality `Single`).

Response:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP01">
        <id>0</id>
        <name>Enter Open Date:</name>
        <answer constrained="false" type="DateTime">
            <info cardinality="Single" keepLastValues="true">
                <previous>
                    <value>1992-09-03T17:15:00.000+02:00</value> <!-- previous
value -->
                </previous>
            </info>
            <values>
                <value>1992-09-03T17:15:00.000+02:00</value> <!-- previous value
-->
            </values>
        </answer>
    </parameter>
</parameters>
```

Example: Prompt with Multiple Values

The parameter of type `prompt` accepts multiple answer values (cardinality `Multiple`).

Response:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>0</id>
        <technicalName>Enter values for Customer:</technicalName>
        <name>Enter values for Customer:</name>
        <answer constrained="false" type="Text">
            <info cardinality="Multiple" keepLastValues="true">
                <lov hierarchical="false" partial="true" refreshable="true"
searchable="false">
                    <id>UNIVERSELOV_DS0.DOd</id>
                    <intervals>
                        <interval id="0">
                            <value>Arai</value>
                            <value>Okumura</value>
                        </interval>
                        <interval id="1">
                            <value>Oneda</value>
```

```
                        <value>Wilson</value>
                    </interval>
                </intervals>
                <values>
                    <value>Arai</value>  <!-- possible values -->
                    <value>Baker</value>
                    <value>Brendt</value>
                    . . .
                    <value>Okumura</value>
                </values>
                <columns mappingId="0">
                    <column id="0" type="String">Customer </column>
                </columns>
            </lov>
            <previous>
                <value>Arai</value>  <!-- previous values -->
                <value>Baker</value>
                <value>Brendt</value>
                <value>Diemers</value>
                <value>Dupont</value>
                <value>Durnstein</value>
                <value>Edwards</value>
            </previous>
        </info>
        <values>
            <value>Arai</value>  <!-- previous values -->
            <value>Baker</value>
            <value>Brendt</value>
            <value>Diemers</value>
            <value>Dupont</value>
            <value>Durnstein</value>
            <value>Edwards</value>
        </values>
    </answer>
</parameter>
. . .
</parameters>
```

> **Note**
>
> `<intervals/>` are not included in the details of the list of values when there is only one interval.

Example: Prompt with Multiple Columns

The parameter of type `prompt` accepts one answer value (cardinality `Single`) from a list of values made of three columns. The first one (`mappingId="0"`) is used as reference.

Response:

```
<parameter optional="false" type="prompt" dpId="DP15">
    <id>6</id>
    <technicalName>Enter values for Customer:</technicalName>
    <name>Enter values for Customer:</name>
    <answer constrained="false" type="Text">
        <info cardinality="Single" keepLastValues="true">
            <lov hierarchical="false" partial="false" refreshable="true"
searchable="true">
                <id>UNIVERSELOV_DS2.DOea</id>
                <intervals>
                    <interval id="0">
                        <cvalue>
```

```
                            <column id="0">William</column>
                            <column id="1">64</column>
                            <column id="2">1995-12-19T02:00:00.000+01:00</column>
                        </cvalue>
                        <cvalue>
                            <column id="0">Silke</column>
                            <column id="1">63</column>
                            <column id="2">1994-03-25T02:00:00.000+01:00</column>
                        </cvalue>
                    </interval>
                    <interval id="1">
                        . . .
                    </interval>
                </intervals>
                <cvalues>
                    <cvalue>
                        <column id="0">Werner</column>
                        <column id="1">42</column>
                        <column id="2">1995-06-08T08:28:00.000+02:00</column>
                    </cvalue>
                    . . .
                    <cvalue>
                        <column id="0">Tony</column>
                        <column id="1">55</column>
                        <column id="2">1995-07-05T04:00:00.000+02:00</column>
                    </cvalue>
                </cvalues>
                <columns mappingId="0">
                    <column id="0" type="String">Customer</column>
                    <column id="1" type="Numeric">Age</column>
                    <column id="2" type="DateTime">Invoice Date</column>
                </columns>
            </lov>
            <previous>
                <value>Andre</value>
            </previous>
        </info>
        <values>
            <value>Andre</value>
        </values>
    </answer>
</parameter>
```


Example: Prompt with Formatted Values

`GET /documents/6874/parameters?formattedValues=true`

`X-SAP-PVL`: `fr_FR`

Response:


```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>0</id>
        <technicalName>Enter value(s) for Col1</technicalName>
        <name>Enter value(s) for Col1</name>
        <answer constrained="false" type="Date">
            <info cardinality="Multiple" keepLastValues="true">
                <lov hierarchical="false" partial="false" refreshable="true"
searchable="true" mandatorySearch="false">
                    <id>UNIVERSELOV_DS0.DO1</id>
                    <updated>2016-01-15T09:41:46.000+01:00</updated>
                    <values>
```

```
                        <value>11/11/1939 00:00:00</value>
                        <value>05/08/1945 00:00:00</value>
                        <value>09/04/1955 00:00:00</value>
                        <value>10/04/1965 00:00:00</value>
                        <value>12/05/1968 00:00:00</value>
                        <value>28/02/1974 00:00:00</value>
                        <value>01/05/1975 00:00:00</value>
                        <value>02/02/1980 00:00:00</value>
                        <value>30/09/1999 00:00:00</value>
                        <value>15/08/2002 00:00:00</value>
                    </values>
                    <columns mappingId="0">
                        <column id="0" type="DateTime">Col1</column>
                    </columns>
                </lov>
                <previous>
                    <value>10/04/1965 00:00:00</value>
                    <value>28/02/1974 00:00:00</value>
                </previous>
            </info>
            <values>
                <value>10/04/1965 00:00:00</value>
                <value>28/02/1974 00:00:00</value>
            </values>
        </answer>
    </parameter>
</parameters>
```


Example: Get a Summary of the Previous Values (Since 4.2 SP04)

```
GET /documents/26647/parameters?summary=true
```

Response:


```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>0</id>
        <technicalName>Year (with default)</technicalName>
        <name>Year (with default)</name>
        <answer constrained="true" type="Text">
            <info cardinality="Single" keepLastValues="false">
                <lov hierarchical="false" refreshable="true"
searchScopes="Values" searchTargets="Server">
                    <id>UNIVERSELOV_DS0.DObc</id>
                </lov>
                <values>
                    <value>2004</value>
                </values>
            </info>
            <values>
                <value>2004</value>
            </values>
        </answer>
    </parameter>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>1</id>
        <technicalName>Quarter (with previous)</technicalName>
        <name>Quarter (with previous)</name>
        <answer constrained="true" type="Text">
            <info cardinality="Single" keepLastValues="true">
                <lov hierarchical="false" refreshable="true"
searchScopes="Values" searchTargets="Server">
                    <id>UNIVERSELOV_DS0.DOba</id>
```

```
                </lov>
                <previous>
                    <value>Q3</value>
                </previous>
            </info>
            <values>
                <value>Q3</value>
            </values>
        </answer>
    </parameter>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>2</id>
        <technicalName>Month (with previous + default)</technicalName>
        <name>Month (with previous + default)</name>
        <answer constrained="true" type="Numeric">
            <info cardinality="Single" keepLastValues="true">
                <lov hierarchical="false" refreshable="true"
searchScopes="Values" searchTargets="Server">
                    <id>UNIVERSELOV_DS0.DOb9</id>
                </lov>
                <values>
                    <value>7</value>
                </values>
                <previous>
                    <value>10</value>
                </previous>
            </info>
            <values>
                <value>10</value>
            </values>
        </answer>
    </parameter>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>3</id>
        <technicalName>Week</technicalName>
        <name>Week</name>
        <answer constrained="false" type="Numeric">
            <info cardinality="Single" keepLastValues="false">
                <lov hierarchical="false" refreshable="true"
searchScopes="Values" searchTargets="Server">
                    <id>UNIVERSELOV_DS0.DO104</id>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```


> **Note**
>
> The `<lovInfo>` optional URL parameter is automatically forced to `false`.

> **Note**
>
> `<sapVariables>` is not set as prompts won't be returned.

Related Information

Getting the List of Documents [page 226]

8.10.2 Refreshing a Document

Usage

Refreshes a Web Intelligence document by filling the refresh parameters if needed and running the query.

You can ask for the refresh without providing any parameters (no request body). In this case, the web service returns the context or prompt that needs to be filled. If no parameter has to be filled, the document is refreshed.

Request

`PUT /documents/<documentID>/parameters?<optional_parameters>`

Where:

Optional Parameters

Parameter	Description
`dataproviderScope`	**Note** New since 4.2 SP04 Optional, String (values="`all`" or "`accessible`". Default value is `all`. If set to `accessible`, a document can be refreshed even if one or more of the data providers is not accessible.
`formattedValues`	Optional, Boolean parameter. Default value is `false`. If set to `true`, the Date-Time and Numeric values are formatted according to the preferred viewing locale set through `X-SAP-PVL` in the request.
`lovInfo`	Optional, Boolean parameter. Default value is `true`. If set to `false`, the lists of values are not computed, nor displayed.
`refresh`	Optional, Boolean parameter. Default value is `true`. To avoid a data refresh at the end of the parameter workflow.
`strict`	Optional, Boolean parameter. Default is `false`. To be used with `<variantIds>`. If set to `true`, an error message is returned in the following cases: • The variant contents and the document parameters do not match • The cardinality of the answer in a variant is incorrect • The data type of the answer in a variant is incorrect
`variantIds`	Optional. The list of variant identifiers separated with a comma that you can use to answer the prompt parameters of the document. The order of variants in the list is important. A value provided by a variant or the request body can overwrite the one provided by another variant. The last value is used to answer the parameter.

Request body: the answers to the parameters retrieved using the `GET .../parameters` call. See Answer Request Body Schemas [page 142]

Response

Response type: `application/xml` or `application/json`

When all parameters have been answered, the last `PUT` call returns a message stating the success of the request.

If `refresh` is `true`, then the message is:

```
<success>
    <message>The resource of type 'Document' with identifier 'XX' has been
successfully updated.</message>
    <id>XX</id>
</success>
```

If `refresh` is `false`, then the message is:

```
<success>
    <message>The resource of type 'Document' with identifier 'XX' has not been
modified.</message>
    <id>XX</id>
</success>
```


Example - Refreshing a Document with a Context [page 627]

Example - Refreshing a Document with Context and Prompt Parameters [page 628]

Example - Refreshing a Document with a Prompt (Formatted Values) [page 629]

Example - Refreshing a Document by Using a Query [page 630]

Example - Refreshing a Document by Using a Query with a Search Pattern [page 631]

Example - Refreshing a Document with a Hierarchical Parameter [page 633]

Example - Refreshing a Document with a Hierarchical Parameter of Multiple Columns [page 635]

Example - Refreshing a Document with Hierarchical Indexed Lists of Values [page 638]

Example - Refreshing a Document with Cascading Parameters [page 642]

Example - Refreshing a Document with Variants [page 644]

Example - Refreshing a Document even if one (or more) of its Data Providers is not Accessible [page 646]

Refresh using the new 4.2 SP04 parameter `dataproviderScope`="accessible".

8.10.2.1 Example - Refreshing a Document with a Context

In the `PUT` call, the request body contains an answer to the context. The following request body does not show the `<info>` element, which is not mandatory:

```
<parameters>
    <parameter optional="false" type="context">
        <id>0</id>
        <name>Select a context</name>
        <answer constrained="true" type="Text">
            <values>
                <value id="2">Reservations</value>
            <values/>
```

```
        </answer>
    </parameter>
</parameters>
```

The PUT call returns a successful message.

```
<success>
    <message>The resource of type 'document' with identifier 'xxx' has been
successfully updated.</message>
    <id>xxx</id>
</success>
```

8.10.2.2 Example - Refreshing a Document with Context and Prompt Parameters

You must first answer the context before answering the subsequent returned prompt. If no context is provided, the document cannot be refreshed, and the response of the PUT call contains the details about the context parameter.

Response:

```
<parameters>
    <parameter optional="false" type="context">
        <id>0</id>
        <name>Select a context</name>
        <answer constrained="true" type="text">
            <info cardinality="Single">
                <values>
                    <value id="2">Reservations</value> <!-- possible values -->
                    <value id="1">Sales</value>
                </values>
                <previous>
                    <value id="1">Sales</value> <!-- previous value -->
                </previous>
            </info>
            <values>
                <value id="1">Sales</value> <!-- previous value -->
            </values>
        </answer>
    </parameter>
</parameters>
```

In a second PUT call, an answer is provided to the context.

Request body:

```
<parameters>
    <parameter optional="false" type="context">
        <id>0</id>
        <answer type="text">
            <values>
                <value id="1">Sales</value>
            </values>
        </answer>
    </parameter>
</parameters>
```

Once the context is resolved, the response of the PUT call contains the previous value for the second parameter of type `prompt`.

Response:

```
<parameters>
    <parameter optional="false" type="prompt">
        <id>1</id>
        <name>Enter a value for Country:</name>
        <answer constrained="false" type="text">
            <info cardinality="Single">
                <previous>
                    <value>France</value>
                </previous>
            </info>
            <values>
                <value>France</value> <!-- previous value -->
            </values>
        </answer>
    </parameter>
</parameters>
```

In another PUT call, the context value and prompt response are given as inputs. Parameters could be in any order, provided that you give all the necessary parameters and their correct identifiers.

Request body:

```
<parameters>
    <parameter optional="false" type="context">
        <id>0</id>
        <answer type="text">
            <values>
                <value id="1">Sales</value>
            </values>
        </answer>
    </parameter>
    <parameter type="prompt">
        <id>1</id>
        <answer type="text">
            <values>
                <value>France</value>
            </values>
        </answer>
    </parameter>
</parameters>
```

Response:

```
<success>
    <message>The resource of type 'document' with identifier '23535' has been
successfully updated.</message>
    <id>23535</id>
</success>
```

8.10.2.3 Example - Refreshing a Document with a Prompt (Formatted Values)

```
PUT /documents/6874/parameters?formattedValues=true
```

```
X-SAP-PVL: en_US
```

Request body:

```
{
  "parameters": {
    "parameter": {
      "id": 0,
      "answer": {
        "values": {
          "value": ["10\/20\/1969 11:45:00 PM"]
        }
      }
    }
  }
}
```

Response:

```
{
 "success":{
   "message":"The resource of type \"Document\" with identifier \"6874\" has
been successfully updated.",
   "id":6874
   }
}
```

8.10.2.4 Example - Refreshing a Document by Using a Query

The request body of the PUT call provides a query that specifies how values should be returned. It does not specify any answer value.

Request body:

```
<parameters>
    <parameter>
        <id>1</id>
        <answer>
            <info>
                <lov>
                    <query intervalId="2" intervalSize="6" refresh="true">
                        <sort order="Descending"/>
                    </query>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

Response:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>1</id>
        <technicalName>Select Max Age</technicalName>
        <name>Select Max Age</name>
        <answer constrained="false" type="Numeric">
            <info cardinality="Single">
                <lov hierarchical="false" partial="false" refreshable="true"
searchable="false">
                    <id>UNIVERSELOV_DS0.DO58</id>
                    <intervals>
```

```
                        <interval id="0">
                            <value>74</value>
                            <value>64</value>
                        </interval>
                        <interval id="1">
                            <value>63</value>
                            <value>45</value>
                        </interval>
                        <interval id="2">
                            <value>42</value>
                            <value>24</value>
                        </interval>
                        <interval id="3">
                            <value>22</value>
                            <value>18</value>
                        </interval>
                    </intervals>
                    <values>
                        <value>42</value>
                        <value>38</value>
                        <value>36</value>
                        <value>34</value>
                        <value>29</value>
                        <value>24</value>
                    </values>
                    <columns mappingId="0">
                        <column id="0" type="double">Age </column>
                    </columns>
                </lov>
                <previous>
                    <value>67</value>
                </previous>
            </info>
            <values>
                <value>67</value>
            </values>
        </answer>
    </parameter>
    . . .
</parameters>
```

8.10.2.5 Example - Refreshing a Document by Using a Query with a Search Pattern

The request body of the PUT call provides a query that specifies which values should be returned according to a pattern. It does not specify any answer value.

Request body (JSON):

```
{"parameters":{
    "parameter":[{
        "id": "1",
        "answer":{
            "info":{
                "lov":{
                    "query":{"search":"20*11??"}
                    }
                }
            }
        }]
    }
```

```
}
```

Response (JSON):


```
{"parameters":{
    "parameter":[
        {...},
        {"@optional":"false",
         "@type":"prompt",
         "@dpId":"DP0",
         "id":1,
         "technicalName":"Enter value(s)",
         "name":"Enter value(s)",
         "answer":{
             "@constrained":"false",
             "@type":"Numeric",
             "info":{
                 "@cardinality":"Multiple",
                 "lov":{
                     "@hierarchical":"false",
                     "@partial":"false",
                     "@refreshable":"false",
                     "@searchable":"true",
                     "@mandatorySearch":"true",
                     "id":"UNIVERSELOV_DS0.DO3",
                     "updated":"2014-12-24T12:51:26.000+01:00",
                     "intervals":{
                         "interval":[
                             {"@id":"0","value":[20121123,20051120]},
                             {"@id":"1","value":[20051121,20071110]},
                             {"@id":"2","value":[20071111,20081130]},
                             {"@id":"3","value":[20111117,20101129]},
                             {"@id":"4","value":[20101130,20091102]}]
                     },
                     "values":{
                         "value":
[20121123,20121124,20121125,20121126,20121127,20121128,20121129,

20121130,20121101,20121102,20121103,20121104,20121105,20121106,

20121107,20121108,20121109,20121110,20121111,20121112,20121113,

20121114,20121115,20121116,20121117,20121118,20121119,20121120,

20121121,20121122,20051101,20051102,20051103,20051104,20051105,

20051106,20051107,20051108,20051109,20051110,20051111,20051112,

20051113,20051114,20051115,20051116,20051117,20051118,20051119,
                                    20051120]
                         },
                     "columns":{
                         "@mappingId":"0",
                         "column":[{
                             "@id":"0",
                             "@type":"Numeric",
                             "$":"Time Key"
                         }]
                     }
                 },
                 "previous":{
                     "value":[20050101]}
             },
         "values":{
             "value":[20050101]}
             }
        }]
```

```
    }
}
```

8.10.2.6 Example - Refreshing a Document with a Hierarchical Parameter

A `GET .../parameters` call returns a list of values for a specific parameter of type `prompt` that asks you to select a city. Values are countries.

The response contains the prompts at the first level in the hierarchy (`hierarchical="true"`). `final="false"` means the value is a node of the hierarchy, while `final="true"` or no tag means value is a leaf.

Response:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DPe">
        <id>0</id>
        <technicalName>Select a city:</technicalName>
        <name>Select a city:</name>
        <answer constrained="true" type="Text">
            <info cardinality="Multiple">
                <lov hierarchical="true" partial="false" refreshable="true"
searchable="false">
                    <id>UNIVERSELOV_DS2.DObb</id>
                    <values>
                        <value final="false">Australia</value>
                        <value final="false">France</value>
                        <value final="false">Germany</value>
                        <value final="false">Holland</value>
                        <value final="false">Japan</value>
                        <value final="false">Madagascar</value>
                        <value final="false">Middle East</value>
                        <value final="false">Nepal</value>
                        <value final="false">South Africa</value>
                        <value final="false">UK</value>
                        <value final="false">US</value>
                    </values>
                    <columns mappingId="0">
                        <column id="0" type="String">Country</column>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

In the first `PUT` call, a value of the first level of the list of values is given as answer to the parameter. Hierarchical lists of values accept an extra `<path>` element in the `<query>` section to go down to each level of the hierarchical parameter. The answer is `France`.

Request body:

```
<parameters>
    <parameter>
        <id>0</id>
        <answer>
            <info>
                <lov>
```

```
                    <query>
                        <path>
                            <value>France</value>
                        </path>
                    </query>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

The response contains the list of values of the second level of the parameter. Values represent regions. The `path` attribute of the `<lov>` element describes the path to any value of the list of values returned in the call. The number represents the data type (0 for string, 1 for date, 2 for numeric).

Response:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DPe">
        <id>0</id>
        <technicalName>Select a city:</technicalName>
        <name>Select a city:</name>
        <answer constrained="true" type="Text">
            <info cardinality="Multiple">
                <lov hierarchical="true" partial="false" refreshable="true"
searchable="false" path="[[0,\ France]]">
                    <id>UNIVERSELOV_DS2.DObb</id>
                    <values>
                        <value final="false">French Alps</value>
                        <value final="false">Normandy</value>
                        <value final="false">Paris</value>
                        <value final="false">Provence</value>
                    </values>
                    <columns mappingId="0">
                        <column id="0" type="String">Region</column>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

In a second `PUT` call, the value of the list of values of the second level is given as answer to the parameter (`Provence`). The first level value is recalled (`France`).

Request body:

```
<parameters>
    <parameter>
        <id>0</id>
        <answer>
            <info>
                <lov>
                    <query>
                        <path>
                            <value>France</value>
                            <value>Provence</value>
                        </path>
                    </query>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

The response contains the list of values of the third and last level of the parameter. Values represent cities.

Response:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DPe">
        <id>0</id>
        <technicalName>Select a city:</technicalName>
        <name>Select a city:</name>
        <answer constrained="true" type="Text">
            <info cardinality="Multiple">
                <lov hierarchical="true" partial="false" refreshable="true"
searchable="false" path="[[0,\ France], [0,\ Provence]]">
                    <id>UNIVERSELOV_DS2.DObb</id>
                    <values>
                        <value final="false">Bordeaux</value>
                        <value final="false">Marseille</value>
                        <value final="false">Nice</value>
                    </values>
                    <columns mappingId="0">
                        <column id="0" type="String">City</column>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

A last `PUT` call refreshes the document with a `<value>`.

Request body:

```
<parameters>
    <parameter>
        <id>0</id>
        <answer>
            <values>
                <value path="[[0,\ France], [0,\ Provence]]">Bordeaux</value>
            </values>
        </answer>
    </parameter>
</parameters>
```

Response:

```
<success>
    <message>The resource of type 'Document' with identifier '9586' has been
successfully updated.</message>
    <id>9586</id>
</success>
```

8.10.2.7 Example - Refreshing a Document with a Hierarchical Parameter of Multiple Columns

A first `GET .../parameters` call returns a list of values for a specific parameter of type `prompt`.

The response contains the prompts at the first level in the hierarchy (`hierarchical="true"`). There is only one column of type String. `final="false"` means the value is a node of the hierarchy, while `final="true"` or no tag means value is a leaf.

Response:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>0</id>
        <technicalName>Enter one or more Invoice Date values</technicalName>
        <name>Enter one or more Invoice Date values</name>
        <answer constrained="false" type="DateTime">
            <info cardinality="Multiple">
                <lov hierarchical="true" partial="false" refreshable="true"
searchable="false">
                    <id>UNIVERSELOV_DS0.DO119</id>
                    <values>
                        <value final="false">FY1992</value>
                        <value final="false">FY1993</value>
                        <value final="false">FY1994</value>
                        <value final="false">FY1995</value>
                    </values>
                    <columns mappingId="0">
                        <column id="0" type="String">Year</column>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

In the first `PUT` call, a value of the first level of the list of values is given as answer to the parameter. Hierarchical lists of values accept an extra `<path>` element in the `<query>` section to go down to each level of the hierarchical parameter. The answer is `FY1993`.

Request body:

```
<parameters>
    <parameter>
        <id>0</id>
        <answer>
            <info>
                <lov>
                    <query>
                        <path>
                            <value>FY1993</value>
                        </path>
                    </query>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

The response contains the list of values of the second level of the parameter. There are two columns of type String. The first one is used as reference (`mappingId="0"`). The `path` attribute of the `<lov>` element describes the path to any value of the list of values returned in the call. The number represents the data type (0 for string, 1 for date, 2 for numeric).

Response:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>0</id>
        <technicalName>Enter one or more Invoice Date values</technicalName>
        <name>Enter one or more Invoice Date values</name>
        <answer constrained="false" type="DateTime">
            <info cardinality="Multiple">
```

```
                <lov hierarchical="true" partial="false" refreshable="true"
searchable="false"
                    path="[[0,\ FY1993]]">
                    <id>UNIVERSELOV_DS0.DO119</id>
                    <cvalues>
                        <cvalue final="false">
                            <column id="0">Q1</column>
                            <column id="1">FY1993</column>
                        </cvalue>
                        <cvalue final="false">
                            <column id="0">Q2</column>
                            <column id="1">FY1993</column>
                        </cvalue>
                        <cvalue final="false">
                            <column id="0">Q3</column>
                            <column id="1">FY1993</column>
                        </cvalue>
                        <cvalue final="false">
                            <column id="0">Q4</column>
                            <column id="1">FY1993</column>
                        </cvalue>
                    </cvalues>
                    <columns mappingId="0">
                        <column id="0" type="String">Quarter</column>
                        <column id="1" type="String">Year</column>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

The `PUT` calls are repeated until the last level of the parameter.

Request body:

```
<parameters>
    <parameter>
        <id>0</id>
        <answer>
            <info>
                <lov>
                    <query>
                        <path>
                            <value>FY1993</value>
                            <value>Q2</value>
                            <value>06</value>
                            <value type="Numeric">25</value>
                        </path>
                    </query>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

There is no more `final="false"` attribute in `<cvalue>` in the response. The last level of the hierarchy is reached, and it contains three columns where the reference is the first one (`mappingId="0"`).

Response:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>0</id>
        <technicalName>Enter one or more Invoice Date values</technicalName>
        <name>Enter one or more Invoice Date values</name>
```

```
        <answer constrained="false" type="DateTime">
            <info cardinality="Multiple">
                <lov hierarchical="true" partial="false" refreshable="true"
searchable="false"
                    path="[[0,\ FY1993], [0,\ Q2], [0,\ 06], [2,\ 25]]">
                    <id>UNIVERSELOV_DS0.DO119</id>
                    <cvalues>
                        <cvalue>
                            <column id="0">1993-06-19T02:00:00.000+02:00</column>
                            <column id="1">06</column>
                            <column id="2">FY1993</column>
                        </cvalue>
                        . . .
                        <cvalue>
                            <column id="0">1995-06-24T02:00:00.000+02:00</column>
                            <column id="1">06</column>
                            <column id="2">FY1995</column>
                        </cvalue>
                    </cvalues>
                    <columns mappingId="0">
                        <column id="0" type="DateTime">Invoice Date</column>
                        <column id="1" type="String">Month</column>
                        <column id="2" type="String">Year</column>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

A last `PUT` call refreshes the document with a `<value>`.

Request body:

```
<parameters>
    <parameter>
        <id>0</id>
        <answer>
            <values>
                <value path='[[0,\ FY1993], [0,\ Q2], [0,\ 06], [2,\
25]]'>1995-06-19T02:00:00.000+02:00</value>
            </values>
        </answer>
    </parameter>
</parameters>
```

Response:

```
<success>
    <message>The resource of type 'Document' with identifier '9586' has been
successfully updated.</message>
    <id>9586</id>
</success>
```

8.10.2.8 Example - Refreshing a Document with Hierarchical Indexed Lists of Values

A `GET .../parameters` call returns a list of values for a specific parameter of type `prompt` and of which lists of values are indexed (`"<value id="...">...</value>"`). Only the value index is mandatory.

The response contains the prompts at the first level in the hierarchy (`hierarchical="true"`). There is only one column of type String. `final="false"` means the value is a node of the hierarchy, while `final="true"` or no tag means value is a leaf.

Response:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>0</id>
        <technicalName>customer Hierarchy Node variable mandatory</technicalName>
        <name>customer Hierarchy Node variable mandatory</name>
        <answer constrained="true" type="Text">
            <info cardinality="Single">
                <lov hierarchical="true" partial="false" refreshable="true"
searchable="false">
                    <id>UNIVERSELOV_DS0.DO48</id>
                    <cvalues>
                        <cvalue id="[Z_COUNTRY COUNTRY_HIERARCHY_01].[WORLD
0HIER_NODE]" final="false">
                            <column id="0">WORLD 0HIER_NODE</column>
                            <column id="1">WORLD</column>
                        </cvalue>
                        <cvalue id="[Z_COUNTRY COUNTRY_HIERARCHY_01].[REST_H
1HIER_REST]" final="false">
                            <column id="0">REST_H 1HIER_REST</column>
                            <column id="1">Not Assigned Country (s)</column>
                        </cvalue>
                    </cvalues>
                    <columns mappingId="0">
                        <column id="0" type="String">LovHierNodeL00 CountryBase
</column>
                        <column id="1" type="String">LovHierNodeL00 Country </
column>
                    </columns>
                </lov>
                <previous>
                    <value id="[Z_COUNTRY COUNTRY_HIERARCHY_01].[0013]">0013</
value>
                </previous>
            </info>
            <values>
                <value id="[Z_COUNTRY COUNTRY_HIERARCHY_01].[0013]">0013</value>
            </values>
        </answer>
    </parameter>
</parameters>
```

In the first `PUT` call, the index of the first level of the list of values is given as answer to the parameter. Hierarchical lists of values accept an extra `<path>` element in the `<query>` section to go down to each level of the hierarchical parameter.

Request body:

```
<parameters>
    <parameter>
        <id>0</id>
        <answer>
            <info>
                <lov>
                    <query>
                        <path>
                            <value id="[Z_COUNTRY COUNTRY_HIERARCHY_01].[WORLD
0HIER_NODE]"/>
                        </path>
                    </query>
```

```
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

The response contains the list of values of the second level of the parameter. The `path` attribute of the `lov` element describes the path to any value of the list of values returned in the call. The syntax `[0|1|2,\ ,\ index]` describes the data type and index of a level (0 for string, 1 for date, 2 for numeric).

Response:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>0</id>
        <technicalName>customer Hierarchy Node variable mandatory</technicalName>
        <name>customer Hierarchy Node variable mandatory</name>
        <answer constrained="true" type="Text">
            <info cardinality="Single">
                <lov hierarchical="true" partial="false" refreshable="true"
searchable="false" path="[[0,\ ,
                    \ [Z_COUNTRY COUNTRY_HIERARCHY_01].[WORLD 0HIER_NODE]]]">
                    <id>UNIVERSELOV_DS0.DO48</id>
                    <cvalues>
                        <cvalue id="[Z_COUNTRY COUNTRY_HIERARCHY_01].[EUROPE
0HIER_NODE]" final="false">
                            <column id="0">EUROPE 0HIER_NODE</column>
                            <column id="1">EUROPE</column>
                        </cvalue>
                        <cvalue id="[Z_COUNTRY COUNTRY_HIERARCHY_01].
[NORTH_AMERICA 0HIER_NODE]" final="false">
                            <column id="0">NORTH_AMERICA 0HIER_NODE</column>
                            <column id="1">NORTH_AMERICA</column>
                        </cvalue>
                        <cvalue id="[Z_COUNTRY COUNTRY_HIERARCHY_01].[ASIA_PAC
0HIER_NODE]" final="false">
                            <column id="0">ASIA_PAC 0HIER_NODE</column>
                            <column id="1">ASIA_PAC</column>
                        </cvalue>
                        <cvalue id="[Z_COUNTRY COUNTRY_HIERARCHY_01].
[MIDDLE_EAST 0HIER_NODE]" final="false">
                            <column id="0">MIDDLE_EAST 0HIER_NODE</column>
                            <column id="1">MIDDLE_EAST</column>
                        </cvalue>
                    </cvalues>
                    <columns mappingId="0">
                        <column id="0" type="String">LovHierNodeL01 CountryBase
</column>
                        <column id="1" type="String">LovHierNodeL01 Country </
column>
                    </columns>
                </lov>
                <previous>
                    <value id="[Z_COUNTRY COUNTRY_HIERARCHY_01].[0013]">0013</
value>
                </previous>
            </info>
            <values>
                <value id="[Z_COUNTRY COUNTRY_HIERARCHY_01].[0013]">0013</value>
            </values>
        </answer>
    </parameter>
</parameters>
```

In a second `PUT` call, the value index of the list of values of the second level is given as answer to the parameter (`"[Z_COUNTRY COUNTRY_HIERARCHY_01].[ASIA_PAC 0HIER_NODE]"`). The first level value index is recalled (`[Z_COUNTRY COUNTRY_HIERARCHY_01].[WORLD 0HIER_NODE]`).

Request body:

```
<parameters>
    <parameter>
        <id>0</id>
        <answer>
            <info>
                <lov>
                    <query>
                        <path>
                            <value id="[Z_COUNTRY COUNTRY_HIERARCHY_01].[WORLD
0HIER_NODE]" />
                            <value id="[Z_COUNTRY COUNTRY_HIERARCHY_01].
[ASIA_PAC 0HIER_NODE]" />
                        </path>
                    </query>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

There is no more `final="false"` attribute in `<cvalue>` in the response. The last level of the hierarchy is reached.

Response:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>0</id>
        <technicalName>customer Hierarchy Node variable mandatory</technicalName>
        <name>customer Hierarchy Node variable mandatory</name>
        <answer constrained="true" type="Text">
            <info cardinality="Single">
                <lov hierarchical="true" partial="false" refreshable="true"
searchable="false" path="[[0,\ ,
                    \ [Z_COUNTRY COUNTRY_HIERARCHY_01].[WORLD 0HIER_NODE]]]"> >
                    <id>UNIVERSELOV_DS0.DO48</id>
                    <cvalues>
                        <cvalue id="[Z_COUNTRY COUNTRY_HIERARCHY_01].[0003]">
                            <column id="0">0003</column>
                            <column id="1">Australia</column>
                        </cvalue>
                        <cvalue id="[Z_COUNTRY COUNTRY_HIERARCHY_01].[0006]">
                            <column id="0">0006</column>
                            <column id="1">Bangladesh</column>
                        </cvalue>
                        <cvalue id="[Z_COUNTRY COUNTRY_HIERARCHY_01].[0015]">
                            <column id="0">0015</column>
                            <column id="1">China</column>
                        </cvalue>
                        ...
                        <cvalue id="[Z_COUNTRY COUNTRY_HIERARCHY_01].[0063]">
                            <column id="0">0063</column>
                            <column id="1">Thailand</column>
                        </cvalue>
                    </cvalues>
                    <columns mappingId="0">
                        <column id="0" type="String">LovHierNodeL02 CountryBase
</column>
```

```
                        <column id="1" type="String">LovHierNodeL02 Country </
column>
                    </columns>
                </lov>
                <previous>
                    <value id="[Z_COUNTRY COUNTRY_HIERARCHY_01].[0013]">0013</
value>
                </previous>
            </info>
            <values>
                <value id="[Z_COUNTRY COUNTRY_HIERARCHY_01].[0013]">0013</value>
            </values>
        </answer>
    </parameter>
</parameters>
```

A last `PUT` call refreshes the document with a `<value>`.

Request body:

```
<parameters>
    <parameter>
        <id>0</id>
        <answer>
            <values>
                <value id="[Z_COUNTRY COUNTRY_HIERARCHY_01].[0015]"/>
            </values>
        </answer>
    </parameter>
</parameters>
```

Response:

```
<success>
    <message>The resource of type 'Document' with identifier '8816' has been
successfully updated.</message>
    <id>8816</id>
</success>
```

8.10.2.9 Example - Refreshing a Document with Cascading Parameters

A first `GET .../parameters` call returns:

- The list of possible values for the answer to a parameter of `id=0`, type `prompt` and cardinality `Single` (age)
- A second parameter of `id=1`, type `prompt`, and cardinality `Single` (customer name)

The first parameter needs to be answered so that you can get the list of values of the second parameter. See the `<parameters>` element inside the `<lov>` element of the second parameter.

Response:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP11">
        <id>0</id>
        <technicalName>Age:</technicalName>
        <name>Age:</name>
        <answer constrained="false" type="Numeric">
```

```
            <info cardinality="Single">
                <lov hierarchical="false" partial="false" refreshable="true"
searchable="false">
                    <id>UNIVERSELOV_DS2.DO9</id>
                    <values>
                        <value>18</value>
                        <value>19</value>
                        . . .
                        <value>150</value>
                    </values>
                    <columns mappingId="0">
                        <column id="0" type="Numeric">Age</column>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
    <parameter optional="false" type="prompt" dpId="DP11">
        <id>1</id>
        <technicalName>Enter customer:</technicalName>
        <name>Enter customer:</name>
        <answer constrained="false" type="Text">
            <info cardinality="Single">
                <lov hierarchical="false" refreshable="true">
                    <id>UNIVERSELOV_DS2.DO10c</id>
                    <parameters>
                        <id>0</id>
                    </parameters>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

A first `PUT .../parameters` call answers the first parameter.

Request body:

```
<parameters>
    <parameter>
        <id>0</id>
        <answer>
            <values>
                <value>25</value>
            </values>
        </answer>
    </parameter>
</parameters>
```

The response contains the possible values for the second parameter.

Response:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP11">
        <id>1</id>
        <technicalName>Enter customer:</technicalName>
        <name>Enter customer:</name>
        <answer constrained="false" type="Text">
            <info cardinality="Single">
                <lov hierarchical="false" partial="false" refreshable="true"
searchable="false">
                    <id>UNIVERSELOV_DS2.DO10c</i>
                    <values>
                        <value>Arrow</value>
                        <value>Baker</value>
```

```
                        . . .
                        <value>Wilson</value>
                    </values>
                    <columns mappingId="0">
                        <column id="0" type="String">Customer</column>
                    </columns>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

> **Note**
>
> Resolving only some of the nested parameters of your document does not allow the main list of values to be filled in with values. If a request body only contains answers to some parameters, the PUT call will return the parameters that need an answer.

8.10.2.10 Example - Refreshing a Document with Variants

The document the user is working with contains five parameters of type prompts and two variants.

First call:

```
GET /documents/9484/parameters
```

Response:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>0</id>
        <technicalName>Choose a Category to Analyze:</technicalName>
        ...
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>1</id>
        <technicalName>Enter Values for State:</technicalName>
        ...
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>2</id>
        <technicalName>Enter Values for Color:</technicalName>
        ...
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>3</id>
        <technicalName>Enter a Value for Year (start):</technicalName>
        ...
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>4</id>
        <technicalName>Enter a Value for Year (end):</technicalName>
        ...
</parameters>
```

Second and third calls:

```
GET /documents/9484/variants/9488
```

The `variant 1` provides value sets for the parameters identified by `0`, `1`, and `2`.

Response:

```
<variant>
```

```
    <id>9488</id>
    <cuid>Abrlk6YtLTpKmlvae2nF3cU</cuid>
    <name>Variant 1</name>
    <parameters>
        <parameter type="prompt" dpId="DP0">
            <technicalName>Choose a Category to Analyze:</technicalName>
            ...
        <parameter type="prompt" dpId="DP0">
            <technicalName>Enter Values for State:</technicalName>
            ...
        <parameter type="prompt" dpId="DP0">
            <technicalName>Enter Values for Color:</technicalName>
            ...
    </parameters>
</variant>
```

`GET /documents/9484/variants/9490`

Response:

The `variant 2` provides value sets for the parameters identified by `2` and `3`.

```
<variant>
    <id>9490</id>
    <cuid>AXFuygctuQJHkwFuwIGz7tc</cuid>
    <name>variant 2</name>
    <parameters>
        <parameter type="prompt" dpId="DP0">
            <technicalName>Enter Values for Color:</technicalName>
            ...
        <parameter type="prompt" dpId="DP0">
            <technicalName>Enter a Value for Year (start):</technicalName>
            ...
    </parameters>
</variant>
```

The following call responds to the prompt parameters of the document:

- By providing the group of value sets from the two variants as query parameters
- By sending values through the request body

`PUT /documents/9484/parameters?variantIds=9488,9490`

Request body:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>1</id>
        <technicalName>Enter Values for State:</technicalName>
        <answer constrained="false" type="Text">
            <values>
                ...
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>4</id>
        <technicalName>Enter a Value for Year (end):</technicalName>
        <answer constrained="false" type="Text">
            <values>
                ...
</parameters>
```

In that case, the parameters are answered as follows:

- `0` through `variant 1`
- `1` through the answer in the request body. This value overwrites the one provided by `variant 1`.

- 2 through the `variant 2`. This value overwrites the one provided by `variant 1`.
- 3 through the `variant 2`
- 4 through the answer in the request body

Finally, use the `strict` query parameter in a call to raise the following HTTP 400 errors in variants:

```
PUT /documents/6861/parameters?variantIds=9330&strict=true
```

- Unused parameter in variant

```
<error>
    <error_code>WSR 00101</error_code>
    <message>Illegal argument (Suspicious variant(s) parameter(s): [{DP13,
'Select Max Age', Numeric, Single}, {DP15, 'Select Your Country', Text,
Multiple}, {DP16, 'Select Your Country', Text, Single}, {DP18, 'Choose
Region', Text, Single}, {DP19, 'Select a customer Date Reservation',
DateTime, Single}, {DP1c, 'Select a customer Date Reservation', DateTime,
Multiple}, {DP1f, 'Select a customer Age', Numeric, Single}])</message>
</error>
```

- Cardinality mismatch

```
<error>
    <error_code>WSR 00101</error_code>
    <message>The variant used to answer the parameter with id '0' has an
incorrect cardinality 'Multiple'. Expected cardinality is 'Single'.</message>
</error>
```

- Data type mismatch

```
<error>
    <error_code>WSR 00101</error_code>
    <message>The variant used to answer the parameter with id '0' has an
incorrect data type 'Text'. Expected data type is 'Numeric'.</message>
</error>
```

8.10.2.11 Example - Refreshing a Document even if one (or more) of its Data Providers is not Accessible

Refresh using the new 4.2 SP04 parameter `dataproviderScope`="accessible".

Usage

Refreshes a Web Intelligence document even if a data provider is not accessible.

Prior to 4.2 SP04

Trying to refresh a document of which one or more of its data providers cannot be accessed, yields to an error. For example `PUT /documents/<12956> /parameters`

Response:

```
<error>
   <error_code>WSR 00402</error_code>
   <message>Resource is not reachable: "You do not have access to one or more
data providers, only the ones for which you have permission will be refreshed.
(Error: WIS 30286)"</message>
   <id>12956</id>
</errorr>
```

Since 4.2 SP04

Trying to refresh a document of which one or more of its data providers cannot be accessed, means that all accessible data providers can be used to refresh the document. For example `PUT /documents/<12956> /parameters?dataproviderScope="accessible"`

Response:

```
<success>
   <message>The resource of type "Document" with identifier "12956" has been
successfully updated.</message>
   <id>12956</id>
</success>
```

> **Note**
>
> Since 4.2 SP4, a new "allDataprovidersRefreshed" property is returned on document refresh success, regardless of the "dataproviderScope" value.

8.10.3 Cancelling the Refresh of a Document

Usage

Cancels the refresh of a Web Intelligence document that is being refreshed. If no execution is currently running, this has no effect.

Since 4.2 SP4, a cancelled 'refresh document' call returns an extra information to warn the user a 'cancel' call has been made.

Request

`PUT /documents/<documentID>/parameters/execution?cancel=<mode>`

Where:

- `<mode>` defines how the cancel is managed:

- `partial`: when the cancel is performed, this displays the new values retrieved so far in the appropriate parts of the document. The rest of the document will display the values retrieved the last time the query was run.
- `restore`: when the cancel is performed, this restores the values to the document that were retrieved the last time the query was run. The values displayed will not be the most up to data information available on the database. You can run the query later to return the up to date values from the database.
- `purge`: when the cancel is performed, this displays the document empty of values. The structure and formatting of the document is retained. You can run the query later to return the up to date values from the database.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/8022/parameters/execution?cancel=partial
```

```
<success>
    <message>The resource of type "Document" with identifier "8022" has been
successfully updated.</message>
    <id>8022</id>
</success>
```

Example: Cancel operation since 4.2 SP04

```
PUT /documents/7395/parameters/execution?cancel=partial
```

```
<success>
    <message>The resource of type "Document" with identifier "8022" has been
successfully updated.</message>
    <id>7935</id>
    <details>
             <property key="cancel">partial</property>
    </details>
</success>
```

Related Information

Getting the List of Documents [page 226]

8.10.4 Getting the Details of a Parameter

Usage

Returns the details of a parameter.

Request

```
GET /documents/<documentID>/parameters/<parameterID>?
formattedValues=<formattedValues>
```

Where:

- `<formattedValues>` is an optional, Boolean parameter. Default value is `false`. If set to `true`, the DateTime and Numeric values are formatted according to the preferred viewing locale set through `X-SAP-PVL` in the request.
- `<lovInfo>` is an optional, Boolean parameter. Default value is `true`. If set to `false`, the LOV (List Of Values) won't be computed thus not displayed. Since 4.2 SP03.
- `<summary>` is an optional, Boolean parameter. Default value is `false`. If set to `true`, a summary of the previous values will be returned. Since 4.2 SP04.

Response

Response type: `application/xml` or `application/json`

The response provides the parameter with its expected answers, previous values if applicable, otherwise default values. See Parameter Response Body Schemas [page 133] to learn about the content structure and element details.

Example: Context

```
GET /documents/6213/parameters/0
```

Response:

```
<parameter optional="false" type="context" dpId="DP1">
    <id>0</id>
    <technicalName>0</technicalName>
    <name>Select a context</name>
    <answer constrained="true" type="Text">
        <info cardinality="Multiple" keepLastValues="true">
            <lov partial="false" searchable="true" mandatorySearch="false">
                <values>
                    <value id="2">Reservations</value>
                    <value id="1">Sales</value>
                </values>
```

```
            </lov>
            <previous>
                <value id="1">Sales</value>
            </previous>
        </info>
        <values>
            <value id="1">Sales</value>
        </values>
    </answer>
</parameter>
```

Example: Get a Summary of the Previous Values

```
GET /documents/26647/parameters?Summary=true
```

Response:

```
<parameters>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>0</id>
        <technicalName>Year (with default)</technicalName>
        <name>Year (with default)</name>
        <answer constrained="true" type="Text">
            <info cardinality="Single" keepLastValues="false">
                <lov hierarchical="false" refreshable="true"
searchScopes="Values" searchTargets="Server">
                    <id>UNIVERSELOV_DS0.DObc</id>
                </lov>
                <values>
                    <value>2004</value>
                </values>
            </info>
            <values>
                <value>2004</value>
            </values>
        </answer>
    </parameter>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>1</id>
        <technicalName>Quarter (with previous)</technicalName>
        <name>Quarter (with previous)</name>
        <answer constrained="true" type="Text">
            <info cardinality="Single" keepLastValues="true">
                <lov hierarchical="false" refreshable="true"
searchScopes="Values" searchTargets="Server">
                    <id>UNIVERSELOV_DS0.DOba</id>
                </lov>
                <previous>
                    <value>Q3</value>
                </previous>
            </info>
            <values>
                <value>Q3</value>
            </values>
        </answer>
    </parameter>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>2</id>
        <technicalName>Month (with previous + default)</technicalName>
        <name>Month (with previous + default)</name>
        <answer constrained="true" type="Numeric">
            <info cardinality="Single" keepLastValues="true">
```

```
                <lov hierarchical="false" refreshable="true"
searchScopes="Values" searchTargets="Server">
                    <id>UNIVERSELOV_DS0.DOb9</id>
                </lov>
                <values>
                    <value>7</value>
                </values>
                <previous>
                    <value>10</value>
                </previous>
            </info>
            <values>
                <value>10</value>
            </values>
        </answer>
    </parameter>
    <parameter optional="false" type="prompt" dpId="DP0">
        <id>3</id>
        <technicalName>Week</technicalName>
        <name>Week</name>
        <answer constrained="false" type="Numeric">
            <info cardinality="Single" keepLastValues="false">
                <lov hierarchical="false" refreshable="true"
searchScopes="Values" searchTargets="Server">
                    <id>UNIVERSELOV_DS0.DO104</id>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

> **Note**
>
> - The lovInfo optional URL parameter is automatically forced to false.
> - sapVariables is not set because prompts won't be returned.

Getting the Details of a Parameter that Depends on Another Parameter

Usage

Returns the details of a parameter of which values depend on the answers of another parameter.

Request

```
PUT /documents/<documentID>/parameters/<parameterID>?
formattedValues=<formattedValues>
```

Where:

- `<formattedValues>` is an optional, Boolean parameter. Default value is `false`. If set to `true`, the DateTime and Numeric values are formatted according to the preferred viewing locale set through `X-SAP-PVL` in the request.

Request body: the details of a parameter on which the parameter you search for depends and a query to return the list of values of the parameter. See Answer Request Body Schemas [page 142] to learn about the content structure and element details.

Response

Response type: `application/xml` or `application/json`

The response provides the parameters with their expected answers, previous values if applicable, otherwise default values. See Parameter Response Body Schemas [page 133] to learn about the content structure and element details.

Example

The details of the parameter "1" depend on the answers to the parameter "3".

`PUT /documents/6713/parameters/1`

Request:

```
<parameters>
    ...
    <parameter>
        <id>3</id>
        <answer>
            <values>
                <value id='2'>Reservations</value>
            </values>
        </answer>
    </parameter>
    <parameter>
        <id>1</id>
        <answer>
            <info>
                <lov>
                    <query intervalSize='Unlimited'/>
                </lov>
            </info>
        </answer>
    </parameter>
</parameters>
```

Response:

```
<parameter optional="false" type="prompt" dpId="DP0">
    <id>1</id>
    <technicalName>Enter values for CustomLOV_withcontext:</technicalName>
    <name>Enter values for CustomLOV_withcontext:</name>
    <answer constrained="false" type="Text">
        <info cardinality="Multiple" keepLastValues="true">
            <lov hierarchical="false" partial="false" refreshable="true"
searchable="true" mandatorySearch="false">
                <id>UNIVERSELOV_DS0.DOda</id>
                <updated>2015-09-08T10:23:04.000+02:00</updated>
                <cvalues>
                    <cvalue>
```

```
                        <column id="0">Abby</column>
                        <column id="1">US</column>
                    </cvalue>
                    <cvalue>
                        <column id="0">Andre</column>
                        <column id="1">Nepal</column>
                    </cvalue>
                    ...
                    <cvalue>
                        <column id="0">Joos</column>
                        <column id="1">Nepal</column>
                    </cvalue>
                </cvalues>
                <columns mappingId="0">
                    <column id="0" type="String">CustomLOV_withcontext</column>
                    <column id="1" type="String">Country</column>
                </columns>
            </lov>
            <previous>
                <value>Abby</value>
                <value>Andre</value>
                <value>Annie</value>
                <value>Bridget</value>
            </previous>
        </info>
        <values>
            <value>Abby</value>
            <value>Andre</value>
            <value>Annie</value>
            <value>Bridget</value>
        </values>
    </answer>
</parameter>
```

Example: With Formatted Values

8.10.5 Working with Variants

A variant consists of a series of values used as answers to some parameters of type prompt in a Web Intelligence document. You can use variants to answer parameters and refresh documents.

Variants are saved in the CMS repository. They are associated with a document and a user. See the *Web Intelligence User Guide* for more information.

Getting the List of Variants [page 654]

Getting the Details of a Variant [page 655]

Creating a Variant [page 660]

Editing a Variant [page 662]

Deleting a Variant [page 663]

Related Information

Refreshing a Document [page 626]

8.10.5.1 Getting the List of Variants

Usage

Returns the list of prompt variants attached to a Web Intelligence document.

Request

```
GET /documents/<documentID>/variants
```

Response

Response type: `application/xml` or `application/json`

Response body: a list of `<variant>` identified by `<id>`, `<cuid>`, and `<name>`.

Example: XML

```
GET /documents/18920/variants
```

Response:

```
<variants>
    <variant>
        <id>18923</id>
        <cuid>AdDOEHufkQZBiims4l6IbW8</cuid>
        <name>my first variant</name>
    </variant>
    <variant>
        <id>18925</id>
        <cuid>Ac55DvETzP1GtMQkmHCFUYw</cuid>
        <name>my second variant</name>
    </variant>
</variants>
```

Example: JSON

Response:

```
{
 "variants": {
     "variant": [
       {
        "id": "18923",
        "cuid": "AdDOEHufkQZBiims4l6IbW8",
        "name": "my first variant"
       },
       {
        "id": "18925",
        "cuid": "Ac55DvETzP1GtMQkmHCFUYw",
        "name": "my second variant"
       }
     ]
 }
}
```

8.10.5.2 Getting the Details of a Variant

Usage

Returns detailed information about a variant created on a Web Intelligence document.

Request

`GET /documents/<documentID>/variants/<variantID>?formattedValues=<formattedValues>`

Where:

- `<formattedValues>` is an optional, Boolean parameter. Default value is `false`. If set to `true`, the DateTime and Numeric values are formatted according to the preferred viewing locale set through `X-SAP-PVL` in the request.

Response

Response type: `application/xml` or `application/json`

The response body provides the following information:

- `<id>`, `<cuid>`, and `<name>` of the variant
- List of parameters of type `prompt` that use the variant
- The values that the variant can take for each parameter

Example: XML

```
GET /documents/18943/variants/18946
```

In this example, the returned variant consists of six parameters of type `prompt`.

Response:

```
<variant>
    <id>18946</id>
    <cuid>AXAUy5cbRbZNjfJ0AV4NPCs</cuid>
    <name>my variant</name>
    <parameters>
        <parameter type="prompt" dpId="DP13">
            <technicalName>Select Max Age</technicalName>
            <answer type="Numeric">
                <info cardinality="Single"/>
                <values>
                    <value>29</value>
                </values>
            </answer>
        </parameter>
        <parameter type="prompt" dpId="DP15">
            <technicalName>Select Your Country</technicalName>
            <answer type="Text">
                <info cardinality="Multiple"/>
                <values>
                    <value>UK</value>
                </values>
            </answer>
        </parameter>
        <parameter type="prompt" dpId="DP16">
            <technicalName>Select Your Country</technicalName>
            <answer type="Text">
                <info cardinality="Single"/>
                <values>
                    <value>US</value>
                </values>
            </answer>
        </parameter>
        <parameter type="prompt" dpId="DP18">
            <technicalName>Choose Region</technicalName>
            <answer type="Text">
                <info cardinality="Single"/>
                <values>
                    <value>Northern Europe</value>
                </values>
            </answer>
        </parameter>
        <parameter type="prompt" dpId="DP19">
            <technicalName>Select a Customer Date Reservation</technicalName>
            <answer type="DateTime">
                <info cardinality="Single"/>
                <values>
                    <value>2016-01-15T01:00:00.000+01:00</value>
                </values>
            </answer>
        </parameter>
        <parameter type="prompt" dpId="DP1c">
            <technicalName>Select a Customer Date Reservation</technicalName>
            <answer type="DateTime">
                <info cardinality="Multiple"/>
                <values>
                    <value>2016-01-19T01:00:00.000+01:00</value>
                    <value>2016-01-25T01:00:00.000+01:00</value>
                    <value>2016-02-08T19:29:00.000+01:00</value>
```

```
                    <value>2016-02-09T01:00:00.000+01:00</value>
                    <value>2016-02-18T01:00:00.000+01:00</value>
                    <value>2016-02-19T05:55:00.000+01:00</value>
                    <value>2016-02-22T01:00:00.000+01:00</value>
                    <value>2016-02-23T01:00:00.000+01:00</value>
                </values>
            </answer>
        </parameter>
        <parameter type="prompt" dpId="DP1f">
            <technicalName>Select a Customer Age</technicalName>
            <answer type="Numeric">
                <info cardinality="Single"/>
                <values>
                    <value>34</value>
                </values>
            </answer>
        </parameter>
    </parameters>
</variant>
```


Example: JSON

Response:


```
{
  "variant": {
    "id": "18946",
    "cuid": "AXAUy5cbRbZNjfJ0AV4NPCs",
    "name": "my variant",
    "parameters": {
      "parameter": [
        {
          "@type": "prompt",
          "@dpId": "DP13",
          "technicalName": "Select Max Age",
          "answer": {
            "@type": "Numeric",
            "info": { "@cardinality": "Single" },
            "values": { "value":["29"]}
          }
        },
        {
          "@type": "prompt",
          "@dpId": "DP15",
          "tecnicalName": "Select Your Country",
          "answer": {
            "@type": "Text",
            "info": { "@cardinality": "Multiple" },
            "values": { "value":["UK"]}
          }
        },
        {
          "@type": "prompt",
          "@dpId": "DP16",
          "technicalName": "Select Your Country",
          "answer": {
            "@type": "Text",
            "info": { "@cardinality": "Single" },
            "values": { "value":["US"]}
          }
        },
        {
          "@type": "prompt",
```

```
          "@dpId": "DP18",
          "technicalName": "Choose Region",
          "answer": {
            "@type": "Text",
            "info": { "@cardinality": "Single" },
            "values": { "value":["Northern Europe"]}
          }
        },
        {
          "@type": "prompt",
          "@dpId": "DP19",
          "technicalName": "Select a Customer Date Reservation",
          "answer": {
            "@type": "DateTime",
            "info": { "@cardinality": "Single" },
            "values": { "value":["2016-01-15T01:00:00.000+01:00"]}
          }
        },
        {
          "@type": "prompt",
          "@dpId": "DP1c",
          "technicalName": "Select a Customer Date Reservation",
          "answer": {
            "@type": "DateTime",
            "info": { "@cardinality": "Multiple" },
            "values": {
              "value": [
                "2016-01-19T01:00:00.000+01:00",
                "2016-01-25T01:00:00.000+01:00",
                "2016-02-08T19:29:00.000+01:00",
                "2016-02-09T01:00:00.000+01:00",
                "2016-02-18T01:00:00.000+01:00",
                "2016-02-19T05:55:00.000+01:00",
                "2016-02-22T01:00:00.000+01:00",
                "2016-02-23T01:00:00.000+01:00"
              ]
            }
          }
        },
        {
          "@type": "prompt",
          "@dpId": "DP1f",
          "technicalName": "Select a Customer Age",
          "answer": {
            "@type": "Numeric",
            "info": { "@cardinality": "Single" },
            "values": { "value":["34"]}
          }
        }
      ]
    }
  }
}
```

Example: XML with Formatted Values

`GET /documents/6915/variants/10789?formattedValues=true`

`X-SAP-PVL`: `fr_FR`

Response:

```
<variant>
```

```
    <id>10789</id>
    <cuid>AUEVb7TLRdNJmUKx2xUe6.g</cuid>
    <name>myVariant</name>
    <parameters>
        <parameter type="prompt" dpId="DP0">
            <technicalName>Enter Reservation Date:</technicalName>
            <answer type="DateTime">
                <info cardinality="Single"/>
                <values>
                    <value>27/12/1983 08:58:12</value>
                </values>
            </answer>
        </parameter>
    </parameters>
</variant>
```

Example: JSON with Formatted Values

`GET /documents/6915/variants/10812?formattedValues=true`

`X-SAP-PVL: en_US`

Response:

```
{
  "variant": {
    "id": 10812,
    "cuid": "ASwjtg1tUTBFjqj7STcQsF0",
    "name": "myVariant",
    "parameters": {
        "parameter":[{
        "@type": "prompt",
        "@dpId": "DP0",
        "technicalName": "Enter Reservation Date:",
            "answer": {
                "@type": "DateTime",
                "info": { "@cardinality": "Single" },
                "values": { "value":["1\/13\/2016 3:11:37 PM"]}
            }
        }]
    }
 }
}
```

Related Information

Getting the List of Variants [page 654]

8.10.5.3 Creating a Variant

Usage

Creates a variant and attaches it to a Web Intelligence document.

Request

`POST /documents/<documentID>/variants?formattedValues=<formattedValues>`

Where:

- `<formattedValues>` is an optional, Boolean parameter. Default value is `false`. If set to `true`, the DateTime and Numeric values are formatted according to the preferred viewing locale set through the `X-SAP-PVL` header in the request.

Request type: `application/xml` or `application/json`

The request body requires at least the following parameter information:

- `type` and `dpId` attributes of `<parameter>`
- `type` value should be `prompt`.
- `<technicalName>` of the parameter
- `type` attribute of `<answer>`
- `cardinality` attribute of `<info>`
- Answer values, of which count depends on the cardinality

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`POST /documents/18954/variants`

Request body:

```
<variant>
    <name>this is a new variant</name>
    <parameters>
        <parameter type="prompt" dpId="DP13">
            <technicalName>Select Max Age</technicalName>
            <answer type="Numeric">
                <info cardinality="Single"/>
```

```
                <values>
                    <value>29</value>
                </values>
            </answer>
        </parameter>
        <parameter type="prompt" dpId="DP15">
            <technicalName>Select Your Country</technicalName>
            <answer type="Text">
                <info cardinality="Multiple"/>
                <values>
                    <value>UK</value>
                </values>
            </answer>
        </parameter>
        <parameter type="prompt" dpId="DP1c">
            <technicalName>Select a Customer Date Reservation</technicalName>
            <answer type="DateTime">
                <info cardinality="Multiple"/>
                <values>
                    <value>2016-01-19T01:00:00.000+01:00</value>
                    <value>2016-01-25T01:00:00.000+01:00</value>
                    <value>2016-02-08T19:29:00.000+01:00</value>
                    <value>2016-02-09T01:00:00.000+01:00</value>
                    <value>2016-02-18T01:00:00.000+01:00</value>
                    <value>2016-02-19T05:55:00.000+01:00</value>
                    <value>2016-02-22T01:00:00.000+01:00</value>
                    <value>2016-02-23T01:00:00.000+01:00</value>
                </values>
            </answer>
        </parameter>
        <parameter type="prompt" dpId="DP1f">
            <technicalName>Select a customer Age</technicalName>
            <answer type="Numeric">
                <info cardinality="Single"/>
                <values>
                    <value>34</value>
                </values>
            </answer>
        </parameter>
    </parameters>
</variant>
```

Response:

```
<success>
    <message>The resource of type "Variant" with identifier "18957" has been
successfully created.</message>
    <id>18957</id>
</success>
```

Example: With Formatted Values

`POST /documents/6915/variants?formattedValues=true`

`X-SAP-PVL`: `en_US`

Request body:

```
<variant>
    <name>myVariant</name>
    <parameters>
        <parameter type="prompt" dpId="DP0">
```

```
            <technicalName>Enter Reservation Date:</technicalName>
            <answer type="DateTime">
                <info cardinality="Single"/>
                <values>
                    <value>1/13/2016 3:11:37 PM</value>
                </values>
            </answer>
        </parameter>
    </parameters>
</variant>
```

Response:

```
<success>
    <message>The resource of type "Variant" with identifier "10812" has been
successfully created.</message>
    <id>10812</id>
</success>
```

8.10.5.4 Editing a Variant

Usage

Modifies the details of a variant attached to a Web Intelligence document.

Request

`PUT /documents/<documentID>/variants/<variantID>?formattedValues=<formattedValues>`

Where:

- `<formattedValues>` is an optional, Boolean parameter. Default value is `false`. If set to `true`, the DateTime and Numeric values are formatted according to the preferred viewing locale set through `X-SAP-PVL` in the request.

Request type: `application/xml` or `application/json`

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/18965/variants/18968
```

Request body:

```
<variant>
    <name>Update Variant</name>
    <parameters>
        <parameter optional="false" type="prompt" dpId="DP13">
            <id>0</id>
            <technicalName>Select Max Age</technicalName>
            <name>Select Max Age</name>
            <answer constrained="false" type="Numeric">
                <info cardinality="Single" keepLastValues="true"/>
                <values>
                    <value>29</value>
                </values>
            </answer>
        </parameter>
        <parameter optional="false" type="prompt" dpId="DP15">
            <id>1</id>
            <technicalName>Select Your Country</technicalName>
            <name>Select Your Country</name>
            <answer constrained="true" type="Text">
                <info cardinality="Multiple" keepLastValues="true"/>
               <values>
                    <value>UK</value>
                </values>
            </answer>
        </parameter>
    </parameters>
</variant>
```

Response:

```
<success>
    <message>The resource of type "Variant" with identifier "18968" has been
successfully updated.</message>
    <id>18968</id>
</success>
```

Related Information

Getting the List of Variants [page 654]

8.10.5.5 Deleting a Variant

Usage

Detaches a variant from a Web Intelligence document and deletes it.

Request

```
DELETE /documents/<documentID>/variants/<variantID>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /documents/18965/variants/18980
```

Response:

```
<success>
    <message>The resource of type "Variant" with identifier "18980" has been
successfully removed.</message>
    <id>18980</id>
</success>
```

Related Information

Getting the List of Variants [page 654]

8.11 Refreshing Data Providers

Below are the refresh operations you can do on a data provider of a Web Intelligence document:

- Refreshing the data provider without contexts and prompts
- Identifying contexts and prompts of a data provider
- Fill in contexts and prompts with values

Supported prompts are the following:

- Prompts that accept either string, numeric or date values
- Prompts that accept one value or multiple values
- Optional and non optional prompts

Getting the Refresh Parameters of a Data Provider [page 665]

Refreshing a Data Provider [page 666]

8.11.1 Getting the Refresh Parameters of a Data Provider

Usage

Returns the parameters to be filled before running a data provider refresh.

Request

```
GET /documents/<documentID>/dataproviders/<dataproviderID>/parameters?
<optional_parameters>
```

Where:

Optional Parameters

Parameter	Description
`formattedValues`	Optional, Boolean parameter. Default value is `false`. If set to `true`, the Date-Time and Numeric values are formatted according to the preferred viewing locale set through `X-SAP-PVL` in the request.
`lovInfo`	Optional, Boolean parameter. Default value is `true`. If set to `false`, the lists of values are not computed, nor displayed.

Response

Response type: `application/xml` or `application/json`

Response body : the parameters with their expected answers, previous values if applicable, otherwise default values. See Parameter Response Body Schemas [page 133]

Example

See Getting the Refresh Parameters of a Document [page 619].

Related Information

Getting the List of Documents [page 226]
Getting the List of Data Providers [page 555]

8.11.2 Refreshing a Data Provider

Usage

Refreshes a Web Intelligence data provider by filling the refresh parameters if needed and running the query.

You can ask for the refresh without providing any parameters (no request body). In this case, the web service returns the context or prompt that needs to be filled. If no parameter has to be filled, the data provider is refreshed.

Request

```
PUT /documents/<documentID>/dataproviders/<dataproviderID>/parameters?
<optional_parameters>
```

Where:

Optional Parameters

Parameter	Description
`formattedValues`	Optional, Boolean parameter. Default value is `false`. If set to `true`, the DateTime and Numeric values are formatted according to the preferred viewing locale set through `X-SAP-PVL` in the request.
`lovInfo`	Optional, Boolean parameter. Default value is `true`. If set to `false`, the lists of values are not computed, nor displayed.

Request body: the answers to the parameters retrieved using the `GET .../parameters` call. See Answer XML Grammar [page 142] to learn about the XML content structure and element details.

Response

Response type: `application/xml` or `application/json`

When all parameters have been answered, the last `PUT` call returns a message stating the success of the request.

```
<success>
    <message>The resource of type "Data provider" with identifier "XX" has been
successfully
        updated.</message>
    <id>XX</id>
</success>
```

Example

See Refreshing a Document [page 626].

Related Information

Getting the List of Documents [page 226]
Getting the List of Data Providers [page 555]

8.12 Scheduling Documents

When you run a schedule, the document is sent in the specified format to some destination at the given time. For more information on scheduling, see the *SAP BusinessObjects Business Intelligence Launch Pad User Guide*.

Getting the List of Schedules [page 667]

Getting the Details of a Schedule [page 669]

Adding a Schedule [page 671]

Deleting a Schedule [page 676]

8.12.1 Getting the List of Schedules

Usage

Gets the list of existing schedules for a Web Intelligence document.

> **Note**
>
> You can find scheduling information in the details of a document. The `<schedules>` element returned in the response body provides `true` if the document has been scheduled, otherwise `false`.

Request

```
GET /documents/<documentID>/schedules
```

Response

Response type: `application/xml` or `application/json`

Response body: the list of schedules for the document, with the following information:

- `<id>`
- `<name>`
- `<format>` type (`webi`, `pdf`, `xls`, `csv`, `csvArchive`, `txt` or `htmlArchive`)
- `<status>` (`Pending`, `Running`, `Paused`, `Completed`, `Warning`, `Expired` or `Failed`)

Example

```
<schedules>
    <schedule>
        <id>28600</id>
        <name>instanceWebi2Inbox</name>
        <format type="webi"/>
        <status id="1">Completed</status>
    </schedule>
    <schedule>
        <id>28609</id>
        <name>instancePDF</name>
        <format type="pdf"/>
        <status id="1">Completed</status>
    </schedule>
    <schedule>
        <id>28651</id>
        <name>instanceWebi2Inbox</name>
        <format type="webi"/>
        <status id="1">Completed</status>
    </schedule>
    <schedule>
        <id>28810</id>
        <name>instanceWebI</name>
        <format type="webi"/>
        <status id="1">Completed</status>
    </schedule>
    <schedule>
        <id>28930</id>
        <name>instanceTXT</name>
        <format type="txt"/>
        <status id="1">Completed</status>
    </schedule>
</schedules>
```

Related Information

Getting the List of Documents [page 226]

8.12.2 Getting the Details of a Schedule

Usage

Gets the details of a schedule of a Web Intelligence document.

Request

```
GET /documents/<documentID>/schedules/<scheduleID>
```

Response

Response type: `application/xml` or `application/json`

Response body:

```
<schedule>
    <id>
    <name>
    <format type="webi|pdf|xls|csv|csvArchive|txt|htmlArchive">
    <deliveryRules>
    <status id="0|1|3|8|9|13|14">
    <updated>
    <destination>
    <error>
    recurrence_expression
    <serverGroup id="integer" required="Boolean"/>
    <parameters>
</schedule>
```

See Schedules [page 152] for a description of the response body.

Example

```
GET /documents/8023/schedules/9439
```

```
<schedule>
    <id>9439</id>
    <name>now-schedule</name>
    <format type="webi"/>
    <status id="1">Completed</status>
    <updated>2015-09-07T08:51:53.214+02:00</updated>
    <destination>
        <ftp>
            <host>vs0202</host>
            <port>21</port>
            <username>admin</username>
            <account/>
            <directory>./</directory>
```

```
        </ftp>
    </destination>
    <once retriesAllowed="0" retryIntervalInSeconds="1800">
        <startdate>2014-06-18T14:19:00.000+02:00</startdate>
        <enddate>2014-06-18T14:20:00.000+02:00</enddate>
    </once>
    <serverGroup id="0" required="false"/>
    <parameters>
        <parameter dpId="DP0">
            <id>1</id>
            <technicalName>Enter value(s) for Year</technicalName>
            <name>Enter value(s) for Year</name>
            <answer>
                <values>
                    <value>2005</value>
                </values>
            </answer>
        </parameter>
    </parameters>
</schedule>
```

In the case of a failure, the response contains an error message. It can be as follows:

```
<schedule>
    <id>9439</id>
    <name>now-schedule</name>
    <format type="webi"/>
    <deliveryRules>
        <deliveryRule>
            <id>ContainsData</id>
            <status>Failed</status>
        </deliveryRule>
    </deliveryRules>
    <status id="3">Failed</status>
    <error>
        <error_code>FWB 00031</error_code>
        <message>Destination disabled. []: [CrystalEnterprise.Ftp]. Please note
the name
            of the job server used for your request and contact your system
administrator
            to make sure the specified destination is enabled. (FWB 00031)</
message>
    </error>
    <destination>
        <ftp>
            ...
        </ftp>
    </destination>
    ...
</schedule>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Schedules [page 667]

8.12.3 Adding a Schedule

Usage

Adds a new schedule to a Web Intelligence document.

A schedule can be run now, once, daily, hourly, weekly, or monthly. A schedule can be triggered by a prompt. You can also select a server group that the system uses to run the schedule.

> **Note**
>
> You can find scheduling information in the document details retrieved from the call `GET /documents/<documentID>`. The `<scheduled>` element returned in the response body provides `true` if the document has been scheduled, otherwise `false`.

Request

`POST /documents/<documentID>/schedules`

Request body: see Schedules [page 152] for a description of the request body

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of a request.

Example - File System Destination for a Now Schedule [page 671]

Example - FTP Destination for a Once Schedule [page 672]

Example - SFTP Destination for an Hourly Schedule [page 672]

Example - Mail Destination for a Hourly Schedule [page 673]

Example - Inbox Destination for a Daily Schedule with a Prompt [page 674]

Example - Inbox Destination for a Monthly Schedule to Specific Users [page 675]

Example - Schedule for a Server Group [page 676]

8.12.3.1 Example - File System Destination for a Now Schedule

This example shows how to schedule "now" a Web Intelligence document in PDF format with file system destination. The document is saved in the `C:\tmp` directory. The default folder is not used.

Example

```
POST /documents/8002/schedules
```

Request body:

```
<schedule>
    <name>nameOfSchedule</name>
    <format type="pdf"/>
    <destination>
        <filesystem>
            <username>user name</username>
            <password>user password</password>
            <directory>C:/tmp/</directory>
        </filesystem>
    </destination>
</schedule>
```

8.12.3.2 Example - FTP Destination for a Once Schedule

This example shows how to run a schedule once of a Web Intelligence document in Microsoft Excel format via FTP.

Example

```
POST /documents/8002/schedules
```

Request body:

```
<schedule>
    <name>nameOfSchedule</name>
    <format type="xls"/>
    <destination>
        <ftp>
            <host>server name</host>
            <port>21</port>
            <username>admin</username>
            <password>admin</password>
            <directory>\server_name\FTP</directory>
        </ftp>
    </destination>
    <once retriesAllowed="2" retryIntervalInSeconds="60">
        <startdate>2012-08-26T15:58:51.000+02:00</startdate>
        <enddate>2013-08-27T15:58:51.000+02:00</enddate>
    </once>
</schedule>
```

8.12.3.3 Example - SFTP Destination for an Hourly Schedule

This example shows how to run an hourly schedule of a Web Intelligence document via SFTP.

Example: With Default Values

```
POST /documents/7037/schedules
```

Default values are taken from the CMC.

Request body:

```
<schedule>
    <name>sftp</name>
    <format type="webi"/>
    <destination>
        <sftp>
            <directory>/tmp/</directory>
        </sftp>
    </destination>
</schedule>
```

Example: With Specific Values

```
POST /documents/7037/schedules
```

The destination details override the default values.

Request body:

```
<schedule>
    <name>my SFTP schedule</name>
    <format type="webi"/>
    <destination>
        <sftp>
            <host>server name</host>
            <port>22</port>
            <username>user name</username>
            <directory>/tmp/</directory>

<fingerprint>c3:52:c6:de:5f:f9:98:fd:05:c8:53:ea:af:64:6b:fb:ad:36:00:47</
fingerprint>
        </sftp>
        <hourly retriesAllowed="4" retryIntervalInSeconds="30">
            <startdate>2015-01-01T14:00:00+02:00</startdate>
            <enddate>2015-12-31T14:00:00+02:00</enddate>
            <hour>12</hour>
            <minute>0</minute>
        </hourly>
    </destination>
</schedule>
```

8.12.3.4 Example - Mail Destination for a Hourly Schedule

This example shows how to run an hourly schedule of a Web Intelligence document in the default format to an email address (every hour between 26/08/2012 and 14/09/2012). If the schedule fails, there will be 2 retries with 60 seconds between each of them.

To schedule a document to an email address over SSL, add the following line to your request body:

```
<mail>
    ...
    <enableSSL>true</enableSSL>
</mail>
```

> → Remember
>
> To schedule to an email address over SSL, make sure you have configured the BI platform properly. See KBA n° 2263613 "How to set up SMTP over SSL".

Example

```
POST /documents/8002/schedules
```

Request body:

```
<schedule>
    <name>nameOfSchedule</name>
    <format type="webi"/>
    <destination>
        <mail>
            <from>somebody@company.com</from>
            <to>person1@company.com;person2@company.com</to>
            <cc>somebody_in_copy@company.com</cc>
            <bcc>list_of_people_in_blind_copy.company.com</bcc>
            <subject>Web Intelligence Restful WS</subject>
            <message>Text to send</message>
            <addAttachment>true</addAttachment>
        </mail>
    </destination>
    <hourly retriesAllowed="2" retryIntervalInSeconds="30">
        <startdate>2012-08-26T14:00:00+02:00</startdate>
        <enddate>2012-09-14T16:00:00+02:00</enddate>
        <hour>1</hour>
        <minute>0</minute>
    </hourly>
</schedule>
```

8.12.3.5 Example - Inbox Destination for a Daily Schedule with a Prompt

This example shows how to run a daily schedule of a Web Intelligence document in the default format to an inbox. By default, the scheduled document is always sent to the sender.

Since the document contains a prompt, the request body also provides the response to the prompt. Prompt text is named "Enter State:" with value "Texas" on a data provider with "DP0" as identifier.

Example

```
POST /documents/8002/schedules
```

Request body:

```
<schedule>
    <name>nameOfSchedule</name>
    <format type="webi"/>
    <destination>
        <inbox/>
    </destination>
    <daily retriesAllowed="2" retryIntervalInSeconds="60">
        <startdate>2012-08-26T15:58:51.000+02:00</startdate>
        <enddate>2012-09-14T15:58:51.000+02:00</enddate>
        <dayinterval>1</dayinterval>
    </daily>
    <parameters>
        <parameter optional="false" type="prompt" dpId="DP0">
            <id>0</id>
            <technicalName>Enter State:</technicalName>
            <answer constrained="false" type="text">
                <values>
                    <value>Texas</value>
                </values>
            </answer>
        </parameter>
    </parameters>
</schedule>
```

8.12.3.6 Example - Inbox Destination for a Monthly Schedule to Specific Users

This example shows how to run a monthly schedule of a Web Intelligence document in the default format to an inbox that specifies four user IDs. The scheduled document is sent as a shortcut by specifying the tag `sendAs` with the value `shortcut`.

Example

```
POST /documents/8002/schedules
```

Request body:

```
<schedule>
    <name>nameOfSchedule</name>
    <format type="webi"/>
    <destination>
        <inbox>
            <to>11,12,1,2</to>
            <sendAs type="shortcut"/>
        </inbox>
    </destination>
    <monthly retriesAllowed="2" retryIntervalInSeconds="60">
        <startdate>2012-08-26T15:58:51.000+02:00</startdate>
```

```
        <enddate>2013-09-14T15:58:51.000+02:00</enddate>
        <month>1</month>
    </monthly>
</schedule>
```

8.12.3.7 Example - Schedule for a Server Group

This example shows how to run a schedule of a Web Intelligence document on servers in a specific server group.

Example

```
<schedule>
    <name>nameOfSchedule</name>
    <format type="webi"/>
    <serverGroup id="6839" required="true"/>
</schedule>
```

8.12.4 Deleting a Schedule

Usage

Deletes a schedule of a Web Intelligence document.

Request

```
DELETE /documents/<documentID>/schedules/<scheduleID>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
<success>
    <message>The resource of type "schedule" with identifier "9372" has been
successfully removed.
    <id>9372</id>
</success>
```

Related Information

Getting the List of Documents [page 226]
Getting the List of Schedules [page 667]

8.13 Managing Publications

A publication is a collection of documents that you send to a mass audience.

Before distributing the documents, you, as the publisher, define the publication using a collection of metadata. The metadata include the publication source, its recipients, and the personalization applied.

8.13.1 Getting the Details of a Publication

Usage

Gets details about an existing publication.

Request

```
GET /publications/<publicationId>
```

Response

Response type: `application/xml` or `application/json`

Example (XML)

```
GET /publications/7412
```

Response:


```
<publication>
    <id>7412</id>
    <cuid>ARHGyoJe1.hEknlXlHI9sj8</cuid>
    <name>pub_multi_docs</name>
    <folderId>6532</folderId>
    <updated>2019-04-05T12:45:42.570Z</updated>
    <createdBy>raylight_user</createdBy>
    <documents>
        <document>
            <id>7001</id>
            <parameters>
                <parameter>
                    <id>1</id>
                    <technicalName>Enter State:</technicalName>
                    <name>Enter State:</name>
                    <answer>
                        <values>
                            <value>New York</value>
                        </values>
                    </answer>
                </parameter>
            </parameters>
        </document>
        <document>
            <id>7145</id>
            <parameters>
                <parameter>
                    <id>1</id>
                    <technicalName>psEnter Res Date:</technicalName>
                    <name>psEnter Res Date:</name>
                    <answer>
                        <values>
                            <value>2010-10-22 00:00:00</value>
                        </values>
                    </answer>
                </parameter>
            </parameters>
        </document>
        <document>
            <id>7239</id>
            <parameters>
                <parameter>
                    <id>1</id>
                    <technicalName>Enter values for Country:</technicalName>
                    <name>Enter values for Country:</name>
                    <answer>
                        <values>
                            <value>Holland</value>
                        </values>
                    </answer>
                </parameter>
            </parameters>
        </document>
        <document>
            <id>7245</id>
        </document>
    </documents>
</publication>
```

8.14 Managing Shared Elements

A shared element is a report part that is stored in the CMS repository and that can be used by multiple users in multiple documents.

APIs are provided that consider shared elements from the following point of views:

- The CMS repository
  You can browse shared element objects, get details, edit and delete them from the CMS repository.
- The Web Intelligence document
  You can get all shared elements inserted into a document and update only one or all of them. You insert a shared element in a document by creating a report element from it.
- The report element in the Web Intelligence document
  You can create a shared element from a report element of a document. You can also browse a shared element linked to a report element and remove the link between them.

See the *SAP BusinessObjects Web Intelligence User's Guide* for more information about shared elements.

Related Information

Shared Element APIs [page 359]

8.14.1 Getting the List of Shared Elements

Usage

Gets the list of all shared elements stored in the CMS repository.

As for any InfoObject, the following items identify a shared element:

- Id
- CUID
- Name
- Description
- Folder Id

Request

```
GET /sharedelements?<optional_parameters>
```

Where:

Optional Parameters

Parameter	Description
`<limit>`	Optional, integer parameter. The maximum number of shared elements to return. The range is [1, 50]. The default value is 10.
`<offset>`	Optional, integer parameter. The offset from the beginning of the list. Default is 0.

Response

Response type: `application/xml` or `application/json`

Response body: the list of shared elements.

Example: XML

```
GET /sharedelements?limit=50&offset=0
```

Response:

```
<sharedelements>
    <sharedelement>
        <id>7691</id>
        <cuid>AT_AVOLuUnBDs.REtpUs6P8</cuid>
        <name>My shared element Cell</name>
        <description>... my description ...</description>
        <folderId>6426</folderId>
    </sharedelement>
    <sharedelement>
        <id>7699</id>
        <cuid>AdcViLuUORtHku0tboAIFW8</cuid>
        <name>My shared element form</name>
        <folderId>6426</folderId>
    </sharedelement>
    <sharedelement>
        <id>7478</id>
        <cuid>AQ8XPJDluEtAnaabTdyERLo</cuid>
        <name>My shared element VTable</name>
        <folderId>6426</folderId>
    </sharedelement>
    <sharedelement>
        <id>7672</id>
        <cuid>AZpwLbwzJSlPp78fAU2xkP0</cuid>
        <name>My shared element HTable</name>
        <folderId>6426</folderId>
    </sharedelement>
</sharedelements>
```

Example: JSON

Response:


```
{
 "sharedelements": {
    "sharedelement": [
    {
     "id": 7691,
     "cuid": "AT_AVOLuUnBDs.REtpUs6P8",
     "name": "My shared element Cell",
     "description": "... my description ...",
     "folderId": 6426
     },
     {
     "id": 7699,
     "cuid": "AdcViLuUORtHku0tboAIFW8",
     "name": "My shared element VTable",
     "folderId": 6426
     },
     {
     "id": 7478,
     "cuid": "AQ8XPJDluEtAnaabTdyERLo",
     "name": "My shared element HTable",
     "folderId": 6426
     },
     {
     "id": 7672,
     "cuid": "AZpwLbwzJSlPp78fAU2xkP0",
     "name": "My shared element Visualization",
     "folderId": 6426
     }
   ]
 }
}
```


8.14.2 Getting the Details of a Shared Element

Usage

Gets the details of a specific shared element, referenced by its ID, in the CMS repository.

Request

`GET /sharedelements/<sharedelementID>`

Response:

Response type: `application/xml` or `application/json`

Response body: the definition of a shared element, with the following information:

- `<id>`
- `<cuid>`
- `<name>`
- `<description>`
- `<folderId>`
- `<path>`
- `<updated>`
- `<createdBy>`
- `<revision>`
- `<keywords>`

Example

```
GET /sharedelements/12446
```

Response (XML):

```
<sharedelement>
    <id>12446</id>
    <cuid>ASBdVOlpZa9ItqHScoGNgDE</cuid>
    <name>my Form SE</name>
    <description>... my description ...</description>
    <folderId>6369</folderId>
    <path>Public Folders/MyDocuments</path>
    <updated>2016-01-11T18:53:12.000+01:00</updated>
    <createdBy>user</createdBy>
    <revision>1</revision>
    <keywords>... my keywords ...</keywords>
</sharedelement>
```

Response (JSON):

```
{
  "sharedelement": {
    "id": "12446",
    "cuid": "ASBdVOlpZa9ItqHScoGNgDE",
    "name": "my Form SE",
    "description": "... my description ...",
    "folderId": "6369",
    "path": "Public Folders/MyDocuments",
    "updated": "2016-01-11T18:53:12.000+01:00",
    "createdBy": "user",
    "revision": "1",
    "keywords": "... my keywords ..."
  }
}
```

Related Information

Getting the List of Shared Elements [page 679]

8.14.3 Editing a Shared Element

Usage

- You can update the following information:
  - Name
  - Description
  - Keywords

The revision number does not change after update, because it is only related to the shared element content.

Request

`PUT /sharedelements/<sharedelementID>`

Request type: `application/xml` or `application/json`

Request body: the relevant information for the element you want to update.

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

`PUT /sharedelements/7691`

Request body:

```
<sharedelement>
    <name>A new name</name>
    <description>A new description</description>
    <keywords>SE1 SE2 (new keywords)</keywords>
</sharedelement>
```

Response body:

```
<success>
    <message>The resource of type "SharedElement" with identifier "7691" has
been successfully updated.</message>
    <id>7691</id>
</success>
```

Related Information

Getting the List of Shared Elements [page 679]

8.14.4 Deleting a Shared Element

Usage

You can delete a shared element from the CMS repository.

Request

```
DELETE /sharedelements/<sharedelementID>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE /sharedelements/7691
```

```
<success>
    <message>The resource of type "SharedElement" with identifier "7691" has
been successfully removed.</message>
    <id>7691</id>
</success>
```

8.14.5 Working with Shared Elements of a Document

APIs are provided to perform tasks on shared elements inserted into a Web Intelligence document.

> **Note**
>
> To insert a shared element into a document, you create a report element from it. See Creating a Report Element [page 391].

8.14.5.1 Getting the List of Shared Elements in a Document

Usage

Gets all shared elements inserted into a Web Intelligence document.

Request

```
GET /documents/<documentID>/sharedelements
```

Response

Response type: `application/xml` or `application/json`

Response body: the list of shared elements with details as below:

- Id
- CUID
- Name
- Folder Id
- Revision number

The revision number may be different of the current revision of the shared element in the CMS repository if it has not been updated.

Example

```
GET /documents/10897/sharedelements
```

Response (XML):

```
<sharedelements>
    <sharedelement>
        <id>10920</id>
        <cuid>Adn7PGNeqWNCoHzW8jy0a7w</cuid>
        <name>my SE VTable</name>
        <folderId>6369</folderId>
        <revision>1</revision>
    </sharedelement>
    <sharedelement>
        <id>12530</id>
        <cuid>AXI5fos7edBPm3xVdTw6nfs</cuid>
        <name>my SE XTable</name>
        <folderId>6350</folderId>
        <revision>3</revision>
    </sharedelement>
</sharedelements>
```

Response (JSON):

```
{
  "sharedelements": {
    "sharedelement": [
      {
        "id": "10920",
        "cuid": "Adn7PGNeqWNCoHzW8jy0a7w",
        "name": "my SE VTable",
        "folderId": "6369",
        "revision": "1"
      },
      {
        "id": "12530",
        "cuid": "AXI5fos7edBPm3xVdTw6nfs",
        "name": "my SE XTable",
        "folderId": "6350",
        "revision": "3"
      }
    ]
  }
}
```

8.14.5.2 Updating a Shared Element of a Document

Usage

Updates a shared element used in the current document with the latest shared element revision stored in the CMS repository.

Request

```
PUT /documents/<documentID>/sharedelements/<sharedelementID>
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/10988/sharedelements/10986
```

Response:

```
<success>
    <message>The resource of type "Document" with identifier "10988" has been
successfully updated.</message>
    <id>10988</id>
</success>
```

8.14.5.3 Updating all Shared Elements of a Document

Usage

Updates all shared elements used in the current document with the latest shared element revisions stored in the CMS repository.

Request

```
PUT /documents/<documentID>/sharedelements
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
PUT /documents/10968/sharedelements
```

Response:

```
<success>
    <message>The resource of type "Document" with identifier "10968" has been
successfully updated.</message>
    <id>10968</id>
</success>
```

8.14.6 Working with Linked Report Elements

8.14.6.1 Saving a Report Element as a Shared Element

Usage

You can create a shared element from a report element of a document. The resulting shared element is stored in the CMS repository and linked to the report element but not inserted into the document.

> **Note**
>
> To insert a shared element into a document, you create a report element from it. See Creating a Report Element [page 391].

You can also update a shared element that already exists in the CMS repository by specifying its name and folder in the request body. The report element content is loaded into the shared element since both elements are linked and the revision number is increased.

Request

```
POST /documents/<documentID>/reports/<reportID>/elements/<elementID>/sharedelement
```

Request type: `application/xml` or `application/json`

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
POST documents/6987/reports/1/elements/15/sharedelement
```

Request:

```
<sharedelement>
    <name>My shared element VTable</name>
    <description>... my description ...</description>
    <keywords>... my keywords ...</keywords>
    <folderId>6426</folderId>
</sharedelement>
```

Response:

```
<success>
    <message>The resource of type "SharedElement" with identifier "7691" has
been successfully created.</message>
    <id>7691</id>
</success>
```

8.14.6.2 Getting the Details of a Shared Element Linked to a Report Element

Usage

Gets the details of a shared element created from a report element of a document and linked to it.

The shared element is inserted into that document and has its own revision number.

Request

```
GET /documents/<documentID>/reports/<reportID>/elements/<elementID>/sharedelement
```

Response

Response type: `application/xml` or `application/json`

Response body: the details of the shared element.

Example

```
GET documents/12593/reports/2/elements/4/sharedelement
```

Response (XML):

```
<sharedelement>
    <id>12591</id>
    <cuid>AYWBcqcvsm5KmmGbR6sZeG4</cuid>
    <name>my shared element XTable</name>
    <folderId>6369</folderId>
    <path>Public Folders/MyDocuments</path>
    <updated>2016-01-11T19:05:42.000+01:00</updated>
    <createdBy>user</createdBy>
    <revision>1</revision>
</sharedelement>
```

Response (JSON):

```
{
  "sharedelement": {
    "id": "12591",
    "cuid": "AYWBcqcvsm5KmmGbR6sZeG4",
    "name": "my shared element XTable",
    "folderId": "6369",
    "path": "Public Folders/MyDocuments",
    "updated": "2016-01-11T19:05:42.000+01:00",
    "createdBy": "user",
    "revision": "1"
  }
}
```

8.14.6.3 Unlink a Shared Element From a Report Element

Usage

Removes the link between a shared element and the report element of a document from which it has been created.

The report element content remains in the document. If the shared element is updated in the CMS repository, the shared element is no longer updated in the document.

Request

```
DELETE /documents/<documentID>/reports/<reportID>/elements/<elementID>/sharedelement
```

Response

Response type: `application/xml` or `application/json`

The response is a message stating the success or failure of the request.

Example

```
DELETE documents/6987/reports/1/elements/15/sharedelement
```

Response:

```
<success>
    <message>The resource of type "Report element" with identifier "15" has been
successfully updated.</message>
    <id>6987</id>
```

```
</success>
```

8.15 Searching for Resources

An API is provided to get interesting resources located on the CMS repository.

In this release, you can search for resources of type folder, universe, connection, Web Intelligence documents, Microsoft Excel files, and CSV files.

Getting Resources [page 691]

8.15.1 Getting Resources

Usage

Returns a list of resources stored on the CMS repository according to a search pattern.

You can search for the following resources:

- Folders
- Universes
- Connections
- Web Intelligence documents
- Microsoft Excel files
- CSV files

The search is targeted to meaningful business intelligence resources, therefore you cannot search for folders only.

Folders, spreadsheets, documents, and CSV files can be searched by identifier. Universes and connections can be searched by resource type and by folder identifier. If no folder identifier is specified, the search is performed on the root folder of the CMS repository.

Request

```
POST /searches?<optional_parameters>
```

Optional Parameters

Parameter	Description
`limit`	Optional, integer parameter. Specifies the maximum number of resources to return per resource type. The range is [0, 1000]. The default value is 1000.

Parameter	Description
`offset`	Optional, integer parameter. Specifies the offset from the beginning of the list. The range is [0, *]. The default value is 0.

Request type: `application/xml` or `application/json`

Request body:

```
<search>
    <folder>
        <folderId>
    <connection>
        <type>
        <folderId>
    <universe>
        <type>
        <folderId>
    <spreadsheet>
        <folderId>
    <document>
        <folderId>
    <csvfile>
        <folderId>
```

Where:

- `<folderId>` is the folder identifier. Is optional.
- `<type>` is the universe type (`unx` or `unv`) or connection type (`Relational`, `Olap`, `FlattenedOlap`, or `DataFederator`). Is optional.

Response

Response type: `application/xml` or `application/json`

Response body: the list of resources requested with the following information:

- Folder ID, CUID, name, and parent folder ID
- Universe ID, CUID, name, type, and folder ID
- Connection type and folder ID
- Document ID, CUID, name, and folder ID
- Microsoft Excel file ID, CUID, name, and folder ID
- CSV file ID, CUID, name, and folder ID

Example: Searching for Universes from the Root Folder (XML)

Request and response types are `application/xml`.

Request body:

```
<search>
```

```
    <universe/>
</search>
```

Response:

```
<search>
    <universe>
        <id>6912</id>
        <cuid>AViz5oUgKhRLrfeDYOPufRk</cuid>
        <name>Refbeach</name>
        <type>unv</type>
        <folderId>532</folderId>
    </universe>
</search>
```

Example: Searching for Universes and Folders from the Root Folder (XML)

Request and response types are `application/xml`.

Request body:

```
<search>
    <folder/>
    <universe/>
</search>
```

Response:

```
<search>
    <folder>
        <id>6011</id>
        <cuid>AVHaHy88GNRHmJCGmL6wBQo</cuid>
        <name>CMSREF_UNIVERSES</name>
        <folderId>532</folderId>
    </folder>
    <folder>
        <id>7619</id>
        <cuid>AZzzdaim0c9IoT9ZFnaimEU</cuid>
        <name>MyUniverses</name>
        <folderId>532</folderId>
    </folder>
    <universe>
        <id>6912</id>
        <cuid>AViz5oUgKhRLrfeDYOPufRk</cuid>
        <name>Refbeach</name>
        <type>unv</type>
        <folderId>532</folderId>
    </universe>
</search>
```

Example: Searching for Folders and Universes in a Specific Folder (JSON)

The search is performed in the folder of ID 7619.

Request and response types are `application/json`.

Request body:

```
{"search":
     {"folder":
         {"folderId":"7619"},
     "universe":
         {"type":"unx", "folderId":"7619"}
     }
}
```

Response:

```
{"search":
     {"folder":
         {"id":"7619", "cuid":"AZzzdaim0c9IoT9ZFnaimEU", "name":"MyUniverses",
"folderId":"532"},
      "universe":[
         {"id":"7777", "cuid":"AbP4G5D9C6BMnOL6k0NzLpk", "name":"ADW.unx",
"type":"unx", "folderId":"7619"},
         {"id":"7784", "cuid":"AdlMl0EJrmNCmQP0UUttgog", "name":"Beach For
Calculations.unx", "type":"unx", "folderId":"7619"}]
     }
}
```

Example: Searching for Folders and Microsoft Excel Files in a Specific Folder (XML)

The search is performed in the folder of ID 6335.

Request and response types are `application/xml`.

Request body:

```
<search>
    <folder>
        <folderId>6335</folderId>
    </folder>
    <spreadsheet>
        <folderId>6335</folderId>
    </spreadsheet>
</search>
```

Response:

```
<search>
    <folder type="Standard">
        <id>6343</id>
        <cuid>Ad87hZEPBJZBg1oLHNBPeVM</cuid>
        <name>Defects</name>
        <folderId>6335</folderId>
    </folder>
    <folder type="Standard">
        <id>6337</id>
        <cuid>Ado_TI0ofM1MoFAuwCgUpqE</cuid>
        <name>Folder1</name>
        <folderId>6335</folderId>
    </folder>
    <spreadsheet>
       <id>6565</id>
       <cuid>AcadmBdXmRFHuV1.EiHMj5M</cuid>
```

```
        <name>CORPDEMO_refreshDP.xlsx</name>
        <folderId>6335</folderId>
    </spreadsheet>
    <spreadsheet>
        <id>6567</id>
        <cuid>AeZCd6SF6LRMp0MV2B5DaS8</cuid>
        <name>Excel_002</name>
        <folderId>6335</folderId>
    </spreadsheet>
</search>
```

8.16 Managing Universes with the Web Intelligence RESTful Web Service SDK

APIs are provided to perform tasks on universes with the help of the SAP BusinessObjects Web Intelligence RESTful Web Service SDK.

Getting the List of Universes (Web Intelligence) [page 695]

Getting the Details of a Universe (Web Intelligence) [page 696]

Getting the Query Capabilities of a Universe (Web Intelligence) [page 700]

8.16.1 Getting the List of Universes (Web Intelligence)

Usage

Gets the list of universes a user has access to, depending on user rights.

Request

`GET /universes?type=<type>&offset=<offset>&limit=<limit>`

Request type: `application/xml` or `application/json`

Query parameters:

- `<type>` indicates the type of the universe. Possible values are `unv`, `unx` and `all`. The default value is `all`. This parameter is optional.
- `<offset>` indicates the position in the list, from which universes are returned. It must be greater than or equal to 0. The default value is 0. This parameter is optional.
- `<limit>` indicates the number of universes that you can list on one page. Its range is [1, 50]. The default value is 10. This parameter is optional.

Response

Response type: `application/xml` or `application/json`

Response body: the list of universes for a user with the following information:

- `<id>`
- `<cuid>`
- `<name>`
- `<type>`
- `<folderId>`

Example

```
GET /universes?type=unx&limit=2
```

```
<universes>
    <universe>
        <id>6773</id>
        <cuid>AXyRzvmRrJxLqUm6_Jbf7lE</cuid>
        <name>efashion.unx</name>
        <type>unx</type>
        <folderId>6771</folderId>
    </universe>
    <universe>
        <id>5612</id>
        <cuid>AYCKrid6ngFGvrKlwVfZKj4</cuid>
        <name>Salary.unv</name>
        <type>unv</type>
        <folderId>509</folderId>
    </universe>
    ...
</universes>
```

8.16.2 Getting the Details of a Universe (Web Intelligence)

Usage

Gets the details of a universe referenced by its ID.

Request

```
GET /universes/<universeID>?aggregated=<aggregated>
```

Where:

- `<aggregated>` is an optional, Boolean parameter that is supported if the universe is `UNX` only. It is ignored if the universe is `UNV`.

The universe details returned depend on the `<aggregated>` value:

- If `true`, the call returns the outline containing all folders and objects granted to the user. This outline merges all granted objects from the granted business view and contains all objects properties such as ID, path, and name.
- If `false`, the call returns the master view if granted or the default view if the master view is denied. The default view name is returned in the outline using the `<businessViewName>` element. This behavior is the one implemented in the SDK versions prior to 4.1 SP5.

If the master view is granted, then the outline returned with the calls `.../<universeID>?aggregated=false` and `.../<universeID>?aggregated=true` are the same, except the `aggregated` outline attribute value.

Response

Response type: `application/xml` or `application/json`

Response body: details of the universe, see Universes [page 157].

Example: The master view is granted

```
GET /universes/9100?aggregated=false
```

or

```
GET /universes/9100
```

```
<universe>
    <id>9100</id>
    <cuid>AftJgs7FPGNBmkdNDnoG8Aw</cuid>
    <name>unx2</name>
    <type>unx</type>
    <folderId>9089</folderId>
    <path>Application Folder/Root Folder/Universes/</path>
    <connected>true</connected>
    <outline aggregated="false">
        <folder>
            <name>City</name>
            <item type="BODimension" dataType="Numeric">
                <name>City Id</name>
                <id>DO1</id>
                <path>City|folder\City Id|dimension</path>
            </item>
            <item type="BODimension" dataType="String">
                <name>City</name>
                <id>DO2</id>
                <path>City|folder\City|dimension</path>
            </item>
            <item type="BODimension" dataType="Numeric">
                <name>Region Id</name>
                <id>DO3</id>
                <path>City|folder\Region Id|dimension</path>
```

```
            </item>
        </folder>
        <folder>
            <name>Country</name>
            <item type="BODimension" dataType="Numeric">
                <name>Country Id</name>
                <id>DO4</id>
                <path>Country|folder\Country Id|dimension</path>
            </item>
            <item type="BODimension" dataType="String">
                <name>Country</name>
                <id>DO5</id>
                <path>Country|folder\Country|dimension</path>
            </item>
        </folder>
        <folder>
            <name>Customer</name>
            <item type="BODimension" dataType="Numeric">
               <name>Cust Id</name>
                <id>DO6</id>
                <path>Customer|folder\Cust Id|dimension</path>
            </item>
            <item type="BODimension" dataType="String">
                <name>First Name</name>
                <id>DO7</id>
                <path>Customer|folder\First Name|dimension</path>
            </item>
            ...
            <item type="BODimension" dataType="Numeric">
                <name>City Id</name>
                <id>DOc</id>
                <path>Customer|folder\City Id|dimension</path>
            ...
        </folder>
        ...
        <folder>
            <name>Reject</name>
            <item type="BODimension" dataType="Numeric">
                <name>Item Id</name>
                <id>DO1e</id>
                <path>Reject|folder\Item Id|dimension</path>
            </item>
            <item type="BODimension" dataType="dateTime">
                <name>Inspection Time</name>
                <id>DO1f</id>
                <path>Reject|folder\Inspection Time|dimension</path>
            </item>
            <item type="BODimension" dataType="String">
                <name>Defect Type</name>
                <id>DO20</id>
                <path>Reject|folder\Defect Type|dimension</path>
            </item>
            <item type="BODimension" dataType="String">
                <name>Nb Rejected</name>
                <id>DO21</id>
                <path>Reject|folder\Nb Rejected|dimension</path>
            </item>
        </folder>
        <folder>
            <name>Inspection</name>
            <item type="BODimension" dataType="Numeric">
                <name>Item Id</name>
                <id>DO22</id>
                <path>Inspection|folder\Item Id|dimension</path>
            </item>
            <item type="BODimension" dataType="dateTime">
                <name>Inspection Time</name>
                <id>DO23</id>
```

```
                <path>Inspection|folder\Inspection Time|dimension</path>
            </item>
            <item type="BODimension" dataType="String">
                <name>Nb Inspected</name>
                <id>DO24</id>
                <path>Inspection|folder\Nb Inspected|dimension</path>
            </item>
        </folder>
        ...
        <folder>
            <name>unv2 Measures</name>
            <item type="Measure" dataType="Numeric">
                <name>Number of Measurement Value</name>
                <id>DO35</id>
                <path>unv2 Measures|folder\Number of Measurement Value|measure</
path>
            </item>
            <item type="Measure" dataType="Numeric">
                <name>Min of Measurement Value</name>
                <id>DDO36</id>
                <path>unv2 Measures|folder\Min of Measurement Value|measure</
path>
            </item>
            ...
            <item type="Measure" dataType="Numeric">
                <name>Number of Nb Rejected</name>
                <id>DO39</id>
                <path>unv2 Measures|folder\Number of Nb Rejected|measure</path>
            </item>
        </folder>
    </outline>
</universe>
```

Example: The master view is denied

```
GET /universes/9100?aggregated=false
```

or

```
GET /universes/9100
```

Response:

```
<universe>
    <id>9100</id>
    <cuid>AftJgs7FPGNBmkdNDnoG8Aw</cuid>
    <name>unx2</name>
    <type>unx</type>
    <folderId>9089</folderId>
    <path>Application Folder/Root Folder/Universes/</path>
    <connected>true</connected>
    <outline aggregated="false">
        <businessViewName>MyView</businessViewName>
        <folder>
            ...
    </outline>
</universe>
```

Example: The call requests an aggregated outline

```
GET /universes/9100?aggregated=true
```

Response:

```
<universe>
    <id>9100</id>
    <cuid>AftJgs7FPGNBmkdNDnoG8Aw</cuid>
    <name>unx2</name>
    <type>unx</type>
    <folderId>9089</folderId>
    <path>Application Folder/Root Folder/Universes/</path>
    <connected>true</connected>
    <outline aggregated="true">
        <folder>
            ...
    </outline>
</universe>
```

Related Information

Getting the List of Universes (Web Intelligence) [page 695]

8.16.3 Getting the Query Capabilities of a Universe (Web Intelligence)

Usage

Gets the query capabilities of a universe.

Request

```
GET /universes/<universeID>/capabilities
```

Response

Response type: `text/xml`

Response body: the following capabilities of a universe, depending on the user rights:

- General query capabilities

- Data processing capabilities
- Filter capabilities: subqueries supported, result hierarchy in filter, object comparison, constant comparison, query on query, and the following lists:
  - The list of supported comparison operators
  - The list of supported logical operators
  - The list of supported object types
  - The list of query on query supported comparison operators
- Result object capabilities

Example

```
GET /universes/2234/capabilities
```

```
<datasource:QueryCapability xmlns:datasource="http://com.sap.sl.datasource">
    <generalCapability combinedQueriesSupported="true"
viewQueryScriptAvailable="true" maxValuesForInList="999"/>
    <dataProcessingCapability removeDuplicateRowsAvailable="true"/>
    <filterCapability subQueriesSupported="true"
resultHierarchyInFilterSupported="false" objectComparisonSupported="true"
        constantComparisonSupported="true" queryOnQuerySupported="true">
        <supportedComparisonOperators>equal</supportedComparisonOperators>
        <supportedComparisonOperators>notEqual</supportedComparisonOperators>
        ...
        <supportedLogicalOperators>or</supportedLogicalOperators>
        ...
        <supportedObjects>attribute</supportedObjects>
        <supportedObjects>dimension</supportedObjects>
        <supportedObjects>measure</supportedObjects>
        ...
        <queryOnQueryCapability>
            <supportedCorrelationTypesByComparisonOperator>
                <value>Any</value>
            </supportedCorrelationTypesByComparisonOperator>
            <supportedCorrelationTypesByComparisonOperator
key="notEqual">
                <value>All</value>
            </supportedCorrelationTypesByComparisonOperator>
            <supportedCorrelationTypesByComparisonOperator key="greater">
                <value>Any</value>
                <value>All</value>
            </supportedCorrelationTypesByComparisonOperator>
            ...
            <supportedCorrelationTypesByComparisonOperator key="between"/>
            <supportedCorrelationTypesByComparisonOperator
key="notBetween"/>
            <supportedCorrelationTypesByComparisonOperator key="inList">
                <value>None</value>
            </supportedCorrelationTypesByComparisonOperator>
            ...
            <supportedComparisonOperatorsByCorrelationType>
                <value>inList</value>
                <value>notInList</value>
            </supportedComparisonOperatorsByCorrelationType>
            <supportedComparisonOperatorsByCorrelationType key="All">
                <value>greater</value>
                <value>greaterOrEqual</value>
                <value>less</value>
                <value>lessOrEqual</value>
                <value>notEqual</value>
```

```
            </supportedComparisonOperatorsByCorrelationType>
            ...
        </queryOnQueryCapability>
        <subQueryCapability>
            <supportedCorrelationTypesByComparisonOperator>
                <value>Any</value>
            </supportedCorrelationTypesByComparisonOperator>
            <supportedCorrelationTypesByComparisonOperator key="notEqual">
                <value>Any</value>
               <value>All</value>
            </supportedCorrelationTypesByComparisonOperator>
            <supportedCorrelationTypesByComparisonOperator key="greater">
                <value>Any</value>
                <value>All</value>
            </supportedCorrelationTypesByComparisonOperator>
            ...
            <supportedCorrelationTypesByComparisonOperator key="isNull"/>
            <supportedCorrelationTypesByComparisonOperator key="notIsNull"/>
            <supportedCorrelationTypesByComparisonOperator key="like"/>
            ...
            <supportedComparisonOperatorsByCorrelationType key="All">
                <value>greater</value>
                <value>greaterOrEqual</value>
               <value>less</value>
                <value>lessOrEqual</value>
                <value>equal</value>
            </supportedComparisonOperatorsByCorrelationType>
            ...
        </subQueryCapability>
    </filterCapability>
    <resultObjectCapability useAttributeSeparatelyAvailable="true"
sortObjectsAvailable="true"
        sortNonResultObjectsAvailable="true"/>
</datasource:QueryCapability>
```

Related Information

Getting the List of Universes (Web Intelligence) [page 695]

8.17 Getting Document Content Through OData

The OData APIs allows you to explore a Web Intelligence document's content and retrieve its dataset.

To get an up-to-date dataset, you may:

- Use the sap_refreshdata=true parameter to refresh the document.
- Schedule the document and use OData to retrieve content from its last instance.

8.17.1 Getting a Document's Cubes and Report Elements

Usage

Get a Web Intelligence document's cubes and report elements through OData.

Request

```
GET documents/cuid_<documentCUID>/datamodel/data.svc
```

Response

Response type: `application/json`

The response returns the list of cubes and reports elements of the document.

8.17.2 Getting a Document's Metadata

Usage

Get a Web Intelligence document's metadata through OData.

Request

```
GET documents/cuid_<documentCUID>/datamodel/data.svc/$metadata
```

Response

Response type: `application/xml`

The response returns the list of cubes and reports elements of the document and their metadata. Its cubes and report elements are exposed as EntityType and their object as EntityType's Property.

8.17.3 Getting a Document's Cube Dataset

Usage

Get a Web Intelligence document's cube dataset through OData.

Request

```
GET documents/cuid_<documentCUID>/datamodel/data.svc/cube_<dataProviderID>-<flowID>
```

This URL can be generated in the Web Intelligence interface. In *Data* mode, in the *Objects* tab, open the contextual menu of a cube and select *Copy OData Web Services Link*. The URL is copied to your clipboard and can be reused in other applications.

The request supports query parameters [page 709] that can be used to modify the query's context.

Response

Response type: `application/json`

The response is a message containing the cube's rows.

8.17.4 Getting a Document's Report Element Dataset

Usage

Get a Web Intelligence document's report element dataset through OData.

Request

```
GET documents/cuid_<documentCUID>/datamodel/data.svc/re_<reportID>-<elementID>
```

This URL can be generated in the Web Intelligence interface. In *Design* mode, right-click a report element. In the contextual menu, select *Copy Link For* > *OData Web Services*. The URL is copied to your clipboard and can be reused in other applications.

The request supports query parameters [page 709] that can be used to modify the query's context.

Response

Response type: `application/json`

The response is a message containing the report element's rows.

8.17.5 Getting a Schedule's Last Instance's Cubes and Report Elements

Usage

Get the list of cubes and report elements of the last instance of a Web Intelligence document's recurring instance through OData.

Request

```
GET documents/cuid_<documentCUID>/schedules/cuid_<scheduleCUID>/datamodel/data.svc
```

This URL can be generated in the SAP BI Launch Pad interface. It can be generated in the *Details* dialog of a recurring instance.

Response

Response type: `application/json`

The response returns the list of cubes and reports elements of last instance of the recurring instance.

8.17.6 Getting a Schedule's Last Instance Metadata

Usage

Get the metadata of the last instance of a Web Intelligence document's recurring instance through OData.

Request

```
GET documents/cuid_<documentCUID>/schedules/cuid_<scheduleCUID>/datamodel/data.svc/
$metadata
```

Response

Response type: `application/xml`

The response returns the list of cubes and reports elements of the last instance of the recurring instance and their metadata. Its cubes and report elements are exposed as EntityType and their object as EntityType's Property.

8.17.7 Getting a Schedule's Last Instance's Cube Dataset

Usage

Get a cube dataset from the last instance of a Web Intelligence document's recurring instance through OData.

Request

```
GET documents/cuid_<documentCUID>/schedules/cuid_<scheduleCUID>/datamodel/data.svc/
cube_<dataProviderID>-<flowID>
```

The request supports query parameters [page 709] that can be used to modify the query's context.

Response

Response type: `application/json`

The response is a message containing the cube's rows in the last instance of the recurring instance.

8.17.8 Getting a Schedule's Last Instance's Report Element Dataset

Usage

Get a report element dataset from the last instance of a Web Intelligence's document recurring instance through OData.

Request

```
GET documents/cuid_<documentCUID>/schedules/cuid_<scheduleCUID>/datamodel/data.svc/
re_<reportID>-<elementID>
```

The request supports query parameters [page 709] that can be used to modify the query's context.

Response

Response type: `application/json`

The response is a message containing the report element's rows in the last instance of the recurring instance.

8.17.9 Getting a Report Element Data Flow Name (Deprecated)

Usage

Get a Web Intelligence report element flow name through OData.

This call is deprecated since SAP BI 4.3 SP3.

Request

```
GET documents/<documentCUID>/reports/<reportID>/elements/<elementID>/data.svc/
$metadata
```

Response

Response type: `application/json`

The response is a message containing the only flow exposed by the report element. This flow is named `Flow0`.

8.17.10 Getting a Report Element Metadata (Deprecated)

Usage

Get a Web Intelligence report element metadata through OData.

This call is deprecated since SAP BI 4.3 SP3.

Request

```
GET documents/<documentCUID>/reports/<reportID>/elements/<elementID>/data.svc/
$metadata
```

Response

Response type: `application/xml`

The response is a message describing the list of objects of the report element and seen as Properties of an EntityType.

8.17.11 Getting a Report Element Data (Deprecated)

Usage

Get a Web Intelligence report element data through OData. This URL can be generated in the Web Intelligence interface.

This call is deprecated since SAP BI 4.3 SP3.

Request

```
GET documents/<documentCUID>/reports/<reportID>/elements/<elementID>/data.svc/Flow0
```

Response

Response type: `application/json`

The response is a message listing the rows of the report element.

8.17.12 Specifying the request

You may add several parameters to the request to modify it and get particular results. The following table describes these possible parameters. These parameters are optional.

Parameter	Response Description
`sap_refreshdata=true`	Refresh the document before returning the query flow.
`sap_user_prompt=(answeredPrompt,…)`	To use with `sap_refreshdata=true` to answer the prompts when the document is refreshed The syntax of an answered prompt is `<promptID in (value1, value2, …)>` where: • String value must be enclosed in double-quote, whose escape code is %27. • Space characters must be escaped by %20. • `<promptID>` is a prompt ID. Prompts IDs are listed under the `EntityType "prompts"` when retrieving document's metadata. For exemple: `sap_user_prompt=( p_x005F_Sales%20in%20(2000), p_x005F_Lines%20in%20(%27Dresses%27,%27Leather%27))` answers two prompts: • `"Sales"` with one value: `2000` • `"Lines"` with two values: `"Dresses"` and `"Leather"`
`$count=true`	The count of records for the specified query.
`$skip=<offset>`	Query results in chunks, starting with the record number specified by the `<offset>` position.
`$top=<limit>`	The first `<limit>` records in chunks of the query flow.
`$skip=<offset>&$top=<limit>`	The first `<limit>` records in chunks, starting with the record number specified by the `<offset>` position in the query flow.
`$select=<object1>,<object2>,…`	Returns only the dataset for the objects specified in the `$select` parameters.

`$orderby=<object> asc`
 	
`$orderby=<object> desc`
	Returns the dataset sorted by the values of the object `<object>`. The sort can be ascending (`asc`) or descending (`desc`).

Parameter	Response Description
`$filter=<filter>`	Filters the returned dataset with the filter specified in `<filter>`. The filter can use the following operators: • Equal: `eq` • Not equal: `ne` • Greater than: `gt` • Lower than: `lt` • In list: `in` • Null: `null` • And: `and` • Or: `or` • Not: `not` For example: `$filter=State ne 'DC' and Month eq 4`

Related Information

Getting a Document's Metadata [page 703]

9 REST API Workflows

API workflows show common use of REST APIs.

Running Queries [page 711]

Creating a Web Intelligence Document [page 713]

Listing Documents [page 714]

Setting Parameters and Exporting to PDF [page 715]

Setting Parameters and Scheduling Now [page 716]

Drilling into a Report [page 717]

9.1 Running Queries

This section explains how you call the REST APIs to create and run queries with or without parameters against universes. The base URL used is `http://<server_name>:6405/biprws/sl/v1`.

Logon

Step	Action	Method	URI	Related Information
1	Logon to the CMS repository and retrieve the authentication information.	GET	/Logon/Long	To Log on to the BI platform [page 51]
2	Retrieve the logon token to be used in next requests.	POST	/Logon/Long	

Simple Query Workflow

Step	Action	Method	URI	Related Information
1	Create the query and send it to the server	POST	/queries	Creating a Query [page 180]

Step	Action	Method	URI	Related Information
2	Run the query	GET	/queries/<queryID>/data.svc	Accessing the OData Service [page 205]
3	Retrieve the query results	GET	/queries/<queryID>/data.svc/<flowName>	Getting the OData Flow Content [page 209]

The GET .../data.svc request actually runs the query on the query engine, accesses the data source, and keeps the result set in memory. Consequently, any further GET call retrieves the cached data without running the query again.

Workflow for a Query With Contexts or Prompts

Step	Action	Method	URI	Related Information
1	Create the query and send it to the server.	POST	/queries	Creating a Query [page 180]
2	Get the query parameters that need to be responded.	GET	/queries/<queryID>/parameters	Getting the List of Parameters [page 184]
3	Respond to the parameters.	PUT	/queries/<queryID>/parameters	Responding to Parameters [page 187]
4	Run the query with the answered parameters.	GET	/queries/<queryID>/data.svc	Accessing the OData Service [page 205]
5	Retrieve the query results.	GET	/queries/<queryID>/data.svc/<flowName>	Getting the OData Flow Content [page 209]

> **Note**
>
> On step 3:
>
> - If all parameters have been answered and the query can be run, then a successful message is returned.
> - Otherwise, the call returns the parameters that remain to be answered. Then, repeat the call until all of the parameters get answers and a success status is returned.

> **→ Remember**
>
> The query execution returns an error message if not all of the parameters have been answered.

Workflow for a Query With Different Parameter Values

You can run the same query mutiple times with different values for contexts or parameters to get different result sets.

Step	Action	Method	URI	Related Information
1	Create the query and send it to the server.	POST	/queries	Creating a Query [page 180]
2	Get the query parameters that need to be responded.	GET	/queries/<queryID>/parameters	Getting the List of Parameters [page 184]
3	Respond to all parameters until all of them have been answered. Repeat the call if necessary.	PUT	/queries/<queryID>/parameters	Responding to Parameters [page 187]
4	Run the query with the answered parameters. The result set is saved in memory.	GET	/queries/<queryID>/data.svc	Accessing the OData Service [page 205]
5	Retrieve the query results from the cache.	GET	/queries/<queryID>/data.svc/<flowName>	Getting the OData Flow Content [page 209]
6	Get the query parameters that need to be responded.	GET	/queries/<queryID>/parameters	Getting the List of Parameters [page 184]
7	Respond to all parameters with new values until all of them have been answered. The cache is reset. Repeat the call if necessary.	PUT	/queries/<queryID>/parameters	Responding to Parameters [page 187]
8	Run the query with the parameters answered in the previous step. The new data set is saved in the cache.	GET	/queries/<queryID>/data.svc	Accessing the OData Service [page 205]
9	Retrieve the query results.	GET	/queries/<queryID>/data.svc/<flowName>	Getting the OData Flow Content [page 209]

> **Note**
>
> Skip step 6 if the parameters have been kept in memory.

Logoff

Step	Action	Method	URI	Related Information
1	Logoff.	POST	/logoff	See the *Business Intelligence Platform RESTful Web Service Developer Guide*.

9.2 Creating a Web Intelligence Document

This section describes the sequence of calls to create a Web Intelligence document that contains a report, and to save it to a specific folder. It is assumed that you know the universe identifier, the dimensions, and attributes

to build the query specification, and how to get the report specification. The base URL used on steps 1, 2, 3 and 12 is `http://<server_name>:6405/biprws`, and from step 4 to step 11 is `http://<server_name>:6405/biprws/raylight/v1`.

Step	Action	Method	URI	Related Information
1	Logon to the CMS repository and retrieve the authentication information.	GET	/Logon/Long	To Log on to the BI platform [page 51]
2	Retrieve the logon token to be used in next requests.	POST	/Logon/Long	
3	Retrieve the document folder ID.	GET	/infostore/cuid_<cuid>	See the *Business Intelligence Platform RESTful Web Service Developer Guide*.
4	Create a Web Intelligence document in the specific folder.	POST	/documents	Creating a Document [page 224]
5	Create a report for the document.	POST	/documents/<documentID>/reports	Creating a Report [page 339]
6	Choose a universe by adding a data provider to the document.	POST	/documents/<documentID>/dataproviders	Adding a Data Provider [page 560]
7	To create your query, add a query specification based on the data provider of the document.	POST	/documents/<documentID>/dataproviders/<dataProviderID>/specification	Updating the Query Specification [page 593]
8	Run your query to get the document data.	PUT	/documents/<documentID>/parameters	Refreshing a Document [page 626]
9	To format your report, add the report structure.	PUT	/documents/<documentID>/reports/<reportID>/specification	Updating the Structure of a Report [page 360]
10	Refresh the document.	PUT	/documents/<documentID>/parameters	Refreshing a Document [page 626]
11	Save the document.	PUT	/documents/<documentID>	Updating the State of a Document [page 247]
12	Logoff.	POST	/logoff	See the *Business Intelligence Platform RESTful Web Service Developer Guide*.

9.3 Listing Documents

The base URL on steps 1, 2 and 5 is `http://<server_name>:6405/biprws`, and on steps 3 and 4 is `http://<server_name>:6405/biprws/raylight/v1`.

Step	Action	Method	URI	Related Information
1	Logon to the CMS repository and retrieve the authentication information.	GET	/Logon/Long	To Log on to the BI platform [page 51]
2	Retrieve the logon token to be used in next requests.	POST	/Logon/Long	
3	Get the first 10 documents sorted alphabetically.	GET	/documents	Getting the List of Documents [page 226]
4	Alternatively, list documents if they are more than 10: 1. Get the first 25 documents sorted alphabetically. 2. Get the next 25 documents.	GET	1. /documents?offset=0&limit=25 2. /documents?offset=1&limit=25	
5	Logoff.	POST	/logoff	See the *Business Intelligence Platform RESTful Web Service Developer Guide*.

9.4 Setting Parameters and Exporting to PDF

The base URL on steps 1, 2 and 9 is `http://<server_name>:6405/biprws`, and on steps 3 to 8 is `http://<server_name>:6405/biprws/raylight/v1`.

Step	Action	Method	URI	Related Information
1	Logon to the CMS repository and retrieve the authentication information.	GET	/Logon/Long	To Log on to the BI platform [page 51]
2	Retrieve the logon token to be used in next requests.	POST	/Logon/Long	
3	List documents to get the ID of a document.	GET	/documents	Getting the List of Documents [page 226]
4	Get details of a document.	GET	/documents/<documentID>	Getting the Details of a Document [page 228]
5	Get the document parameters as a template to be filled.	GET	/documents/<documentID>/parameters	Getting the Refresh Parameters of a Document [page 619]
6	Set the document parameters to refresh the document with new values.	PUT	/documents/<documentID>/parameters	Refreshing a Document [page 626]

Step	Action	Method	URI	Related Information
7	Export each report in the document as a single PDF page	GET	/documents/<documentID> with accept:application/pdf	Exporting a Document in Listing Mode [page 243]
8	Export to PDF as a series of pages including page headers and footers	GET	/documents/<documentID>/pages with accept:application/pdf	Exporting a Document as a Series of Pages [page 245]
9	Logoff.	POST	/logoff	See the *Business Intelligence Platform RESTful Web Service Developer Guide*.

9.5 Setting Parameters and Scheduling Now

The base URL on steps 1, 2 and 9 is http://<server_name>:6405/biprws, and on steps 3 to 8 is http://<server_name>:6405/biprws/raylight/v1.

Step	Action	Method	URI	Related Information
1	Logon to the CMS repository and retrieve the authentication information.	GET	/Logon/Long	To Log on to the BI platform [page 51]
2	Retrieve the logon token to be used in next requests.	POST	/Logon/Long	
3	List documents to get the ID of a document.	GET	/documents	Getting the List of Documents [page 226]
4	Get details of a document.	GET	/documents/<documentID>	Getting the Details of a Document [page 228]
5	Get the document parameters as a template to be filled.	GET	/documents/<documentID>/parameters	Getting the Refresh Parameters of a Document [page 619]
6	Get the list of schedules.	GET	/documents/<documentID>/schedules	Getting the List of Schedules [page 667]
7	Schedule the document now with parameter values. The response contains the new schedule ID.	POST	/documents/<documentID>/schedules	Adding a Schedule [page 671]
8	View schedule instance details.	GET	/documents/<documentID>/schedules/<scheduleID>	Getting the Details of a Schedule [page 669]
9	Logoff.	POST	/logoff	See the *Business Intelligence Platform RESTful Web Service Developer Guide*.

9.6 Drilling into a Report

This section describes the sequence of calls to open a Web Intelligence document and drill through a report data.

The base URL on steps 1, 2 and 17 is `http://<server_name>:6405/biprws`, and on steps 3 to 16 is `http://<server_name>:6405/biprws/raylight/v1`.

Step	Action	Method	URI	Related Information
1	Logon to the CMS repository and retrieve the authentication information.	GET	`/Logon/Long`	To Log on to the BI platform [page 51]
2	Retrieve the logon token to be used in next requests.	POST	`/Logon/Long`	
3	List the available documents.	GET	`/documents`	Getting the List of Documents [page 226]
4	Get the document details.	GET	`/documents/<documentID>`	Getting the Details of a Document [page 228]
5	Get the list of reports of this document.	GET	`/documents/<documentID>/reports`	Getting the List of Reports [page 341]
6	Get the report details.	GET	`/documents/<documentID>/reports/<reportID>`	Getting the Details of a Report [page 342]
7	Enable drill.	POST	`/documents/<documentID>/reports/<reportID>/driller`	Enabling the Query Drill [page 379]
8	Check the drill is enabled.	GET	`/documents/<documentID>/reports/<reportID>/driller`	Getting the Drill Mode [page 377]
9	Show drill elements and whether they are in scope.	GET	`/documents/<documentID>/reports/<reportID>/driller/hierarchies`	Getting Information on the Drill Hierarchies [page 380]
10	Get the drill filters and see that none exists.	GET	`/documents/<documentID>/reports/<reportID>/driller/filters`	Getting the Drill Filters of a Report [page 384]
11	Add a filter.	POST	`/documents/<documentID>/reports/<reportID>/driller/filters`	Creating a Drill Filter [page 383]
12	Get the details of the drill filter to see what values are available.	GET	`/documents/<documentID>/reports/<reportID>/driller/filters/<filterID>`	Getting the Details of a Drill Filter [page 385]

Step	Action	Method	URI	Related Information
13	Drill on a value.	PUT	`/documents/<documentID>/reports/<reportID>/driller/filters/<filterID>`	Updating a Drill Filter [page 386]
14	View HTML content and see that you drilled on a specific filter value.	GET	`/documents/<documentID>/reports/<reportID>` with `accept:text/html`	Exporting a Report in Listing Mode [page 350]
15	Update the drill filter to drill on another value.	PUT	`/documents/<documentID>/reports/<reportID>/driller/filters/<filterID>`	Updating a Drill Filter [page 386]
16	View HTML content and see that you drilled on a specific filter value.	GET	`/documents/<documentID>/reports/<reportID>` with `accept:text/html`	Exporting a Report in Listing Mode [page 350]
17	Logoff.	POST	`/logoff`	See the *Business Intelligence Platform RESTful Web Service Developer Guide*.

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 an 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.

Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

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.

Bias-Free Language

SAP supports a culture of diversity and inclusion. Whenever possible, we use unbiased language in our documentation to refer to people of all cultures, ethnicities, genders, and abilities.

www.sap.com/contactsap

© 2024 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.

THE BEST RUN SAP