UNICOM® Global
Putting IT All Together.℠

Focal Point®

JSON REST API Reference Manual

Release 7.50

UNICOM® Systems, Inc.
A Division of UNICOM Global

Publication information

FPJR-7500-00 (December 2021)

Information in this publication is subject to change. Changes will be published in new editions or technical newsletters.

Copyright notice

Focal Point®

Disclaimer

We cannot guarantee freedom from, or assume any responsibility or liability for technical inaccuracies or typographical errors. The information herein is furnished for informational use only and should not be construed as a commitment by UNICOM Systems, Inc. – a division of UNICOM Global.

Trademarks

The following are trademarks or registered trademarks of UNICOM Systems, Inc. in the United States and/or other jurisdictions worldwide: Focal Point, UNICOM, UNICOM Systems.

For a list of third-party products, companies, trademarks, and registered trademarks that might be referenced in this manual, see *www.unicomsi.com/trademarks*.

Contents

1 JSON REST API reference

The Representational State Transfer (REST) API for Focal Point is a RESTful interface through which you can access, create and update Focal Point resources by using JSON data format.

Setup

To specify the server name for the resource URI, in Focal Point, click **Application > Login Page > Login or Balancer URL** and specify the appropriate value for the host name.

Note Make sure that the host name or the server name does not change. A change in the host name can lead to broken links in Focal Point integrated systems that link to the Focal Point resources.

Authentication

The requests to the RESTful API must be authenticated by using HTTP basic authentication. Unless you use HTTPS authentication, the user name and password are sent without encryption.

In HTTP basic authentication, character encoding is not specified for user names and passwords. User names and passwords can include only ASCII characters. You might be able to use ISO-8859-1 characters if the characters are correctly encoded by the client.

Character encoding

The JSON information that is retrieved from Focal Point uses UTF-8 character encoding.

For JSON data that is sent to Focal Point, the charset of the Content-Type HTTP header is used (if present). Otherwise, the encoding that is specified in the JSON prolog is used. If no prolog is present, the default for the JSON data is used. The encoding that is used must match any declared encoding. You can use UTF-8 encoding when you communicate with Focal Point and specify UTF-8 encoding in the HTTP header and JSON prolog.

Accessing JSON REST APIs in a multi threaded environment

In Focal Point version 6.5 and later, for client applications to access the REST API in a multi-threaded environment, you must follow these steps:

1 Use the MultiThreadedHttpConnectionManager object to create the HTTP connection object. For example:

```
...
MultiThreadedHttpConnectionManager connectionManager = new
MultiThreadedHttpConnectionManager();
client = new HttpClient(connectionManager );
...
```

2 Reuse the HttpClient object from each thread to make REST API requests instead of creating a new HttpClient object for each thread.

Examples

You can refer to the REST API examples to learn how to use the Focal Point APIs. The examples are in Java source files that can be compiled and run.

Prerequisite

Knowledge of Java and REST.

Resources overview

The REST JSON resources in Focal Point are service, element collections, element, and attributes. The following table summarizes the resource types and example REST API URIs for the resources:

Resource type	Example REST API URI	Description	Supported REST operations
Service document	`https://fpserver.com:9443/fp/resources/`	The service document has the high level resources for the workspace, module, views that a user has access to. For detailed information, see *'Service documents' on page 6*.	GET
Element collection	`https://fpserver.com:9443/fp/resources/workspaces/1/modules/1/elements`	An element collection has the contents of a view or module. For detailed information, see *'Element collection' on page 7*.	GET POST PUT
Element	`https://fpserver.com:9443/fp/resources/workspaces/1/modules/1/elements/1`	Element has contents of attributes. For detailed information, see *'Elements' on page 14*.	GET
Attribute	`https://fpserver.com:9443/fp/resources/workspaces/1/modules/1/elements/1/attributes/1`	Attribute. For detailed information, see *'Attributes' on page 16*.	GET POST PUT DELETE

Links

The JSON representations of Focal Point resources can link to other resources by using the JSON element link. The link is similar to the link in the Atom format.

For more information, see *http://www.atomenabled.org/developers/syndication/atom-format-spec.php#element.link*.

Link attribute	Description
href	(mandatory) The URI of the referenced resource.
rel	(optional) The rel attribute can have one of the following values: ▪ **alternate**: An alternate representation of the resource. For example, a HTML page. ▪ **enclosure**: A related resource that might be large and require special handling. For example, a binary file. ▪ **related**: A related resource. ▪ **self**: The resource representing itself. ▪ **edit**: A reference to a resource that can be edited, and might also be the resource.
title	(optional) A title for the link.
type	(optional) The media (MIME) type of the resource.
hreflang	(optional) The language of the resource.
length	(optional) The size of the referenced resource in bytes.

Service documents

The service document has the high level resources for workspace, module, and views that a user has access to. Only users with workspace administrator rights can access modules.

The resource URI for the service document is in the form: `http://fpserver/fp/resources/`

GET method

Accept header: `application/json`

Response body:

{"workspaces":[{"alias":"878914ee-a997-457f-b6ed-25396c8b8d5e","title":"IT Portfolio Management Template - MASTER DEMO","addViews":[{"addable":"true","alias":"84c32448-2138-4eb5-bdc2-d8d75e78b244","indexList":"https://fpserver.com:9443/fp/resources/workspaces/201/modules/14/elements/?view=834","title":"Application","indexTree":"https://fpserver.com:9443/fp/resources/workspaces/201/modules/14/elements/?view=834&tree=true"}],"displayViews":[{"addable":"false","alias":"3920ffde-429d-4714-803d-bec47d23c362","indexList":"https://fpserver.com:9443/fp/resources/workspaces/201/modules/2/elements/?view=838","fullTree":"https://fpserver.com:9443/fp/resources/workspaces/201/modules/2/elements/?view=838&tree=true&includeAttributes=true","title":"Strategic Objectives (r/-)","indexTree":"https://fpserver.com:9443/fp/resources/workspaces/201/modules/2/elements/?view=838&tree=true","fullList":"https://fpserver.com:9443/fp/resources/workspaces/201/modules/2/elements/?view=838&includeAttributes=true"}]}]}

Service document description

Each module and view contains an element collection. The following table shows the ways that you can view the element collection in the service document:

Link	Visualization
indexList	A list of elements that includes only the title and URI of each element. If a sort attribute is defined, the elements are sorted accordingly.
indexTree	A tree structure of elements that includes only the title and URI of each element. The title (but not the URI) of folders that not are displayed is included.
fullList	A list of elements that includes all attributes values for the attributes in the view or module. If a sort attribute is defined, the elements are sorted accordingly.
fullTree	A tree structure of elements that includes the attribute values for the attributes in the view or module.

Element collection

An element collection has the contents of a view or module. The URI for an element collection is available in the service document and the general URI is in the form:

`http://`*`fpserver`*`/context/resources/workspaces/`*`n`*`/modules/`*`n`*`/elements/`

The JSON representation of the element collection contains a URI for each element in the element collection. Depending on the link that was chosen in the service document, the collection might include attribute values and might be structured as tree or a flat list.

A single element contains all attribute values and contains more details than the collection.

If a collection is an Add view or a module, you can add an element to the collection by using a POST operation. The body of the request must contain a JSON representation of the element. If the collection is an Add view, the contents of the view are in the folders. The folders can be parents when a new element is added.

Note RESTful APIs ignore the filters that are applied in views. All the elements of a view are displayed.

GET method

Use the GET method to retrieve the collection of elements in a module. You can use the View parameter to filter the elements and attributes. For example:

URL:
`https://fpserver.com:9443/fp/resources/workspaces/2/modules/1/elements/.json?includeAttributes=true&view=14`

Accept header: `application/json`

Response body:


```
{"elementCollection":[{"elementId":27,"alias":"789944da-2d00-42a7-a4f3-
74610087dcd4","title":"External
Applications","selfLink":"https://fpserver.com:9443/fp/resources/workspaces/49/modules/1/eleme
nts/27?view=427"},{"elementId":26,"alias":"dee607ec-f9a4-4237-a809-
8f92ac67d8a3","title":"Business
Operation","selfLink":"https://fpserver.com:9443/fp/resources/workspaces/49/modules/1/elements
/26?view=427"}],"moduleName":"Portfolios","count":2,"links":[{"rel":"alternate","href":"https:
//fpserver.com:9443/fp/resources/workspaces/49/modules/1/elements/?view=427&includeAttributes=
true&tree=true"},{"href":"https://fpserver.com:9443/fp/resources/workspaces/49/modules/1/eleme
nts/?view=427&includeAttributes=true"},{"href":"https://fpserver.com:9443/fp/resources/workspa
ces/49/modules/1/elements/?view=427&tree=true"}]}
```


Supported parameters for the GET method

The following parameters can be used with the GET method:

- **includeAttributes**

  Set the `includeAttributes` parameter to `true` to retrieve the attributes of the elements.

  Type: Boolean

- **view**

  Use the `view` parameter to specify the view ID for retrieving the elements or attributes that are in the view.

  Type: Integer

- **filter**

  The `filter` parameter must be used with the `view` parameter. Use the `filter` parameter to pass the filter ID for retrieving elements and attributes based on the filter criteria.

  Type: Integer

  For example, `view=14&filter=1`.

- **tree**

  Set the `tree` parameter to `true` to view the element collection in the tree structure

  Type: Boolean

- **modifiedSince**

Use the `modifiedSince` parameter to retrieve the element that has changed since a specified date. The following formats are supported.

Note The user can also supply `modifiedSince` as a parameter.

```
yyyy-MM-dd'T'HH:mm:ss
yyyy-MM-dd'T'HH:mm
yyyy-MM-dd
yyyy-MM-dd HH:mm:ss
yyyy-MM-dd HH:mm
```

Type: Date or DateTime

For example, `2020-04-19T15:16:38+05:30 or 2020-01-22T12:00:00+01:00`.

Note Make sure to encode the value before adding to URL.

- **fields**

  Use the `fields` parameter to pass XPath expression to retrieve a specific set of attributes or to filter elements that are based on specific values of attributes.

  Type: XPath expression

  For example, to retrieve a specific set of attributes, use:

  ```
  fields=elementCollection/Element/attributes/(ID|Title|Owner))
  ```

  or to filter elements based on attribute values, use:

  ```
  fields=elementCollection/Element/attributes/(Title contains X OR Status
  is Approved )
  fields=elementCollection/Element/attributes/(Title =  X | Status is
  Approved )
  fields=elementCollection/Element/attributes/(Title is X AND Status =
  Approved )
  fields=elementCollection/Element/attributes/(Title contains X AND Cost
  < 100 )
  ```

  Notes

  - Separator '|' should be encoded as `'%7C'`.

    For example, `fields=elementCollection/Element/attributes/(ID | Title)` should be encoded as `fields=elementCollection/Element/attributes/(ID %7C Title)`.

  - Attribute names and the attribute values should be enclosed in single quotes if they contain the following special characters: `(`, `)`, <, >, `|`, =.

    For example,
    `fields=elementCollection/Element/attributes/('Comments(incl)' contains 'Shift>Technology')`.

If the names or values already contain single quotes (') then they must be escaped with a backslash as: \'.

- The following characters or character sequences can be used as operators: `=`, `<`, `>`, `is`, `contains`.
- Different filtering conditions can be connected by using the following connectors (case sensitive): `AND`, `OR`, `|` (same as `OR`).
- If you are using the `fields` parameter to filter the elements, do not use the `modifiedSince` parameter as well. Instead, use the `fields` parameter with a filtering condition specified on the Last Changed Date attribute.
- You can specify filtering conditions on only the following attributes: Text, Integer, Float, Date, Choice, Check Box, MulitChoice, Parent Folder, Created Date, Created By, Last Changed Date, Last Changed By, Owner, Link, Linklist, Incoming Link, Unique Id.
- For attributes like Choice and MultiChoice, you can specify the name of the items as the filtering value. For Link, Linklist and Incoming Link, you must specify the id of the target element as the filtering condition. For Parent Folder, Owner, Created By and Last Modified By attributes, you can either specify the name or the id of the target element.
- To filter based on more than one attribute value, provide the filtering condition by comma-separating the individual values.

  For example, the following example filters the elements where ‘Geography’ contains both ‘Asia’ and ‘Europe’ (applies 'all' condition):

  ```
  fields=elementCollection/Element/attributes/(Geography contains Asia,Europe).
  ```

- For multivalued attributes like MultiChoice and Linklist, you can prefix the filtering condition with `or`, to specify the 'any' condition while filtering.

  For example, the following example filters the elements where ‘Geography’ contains either Asia or Europe (applies 'any' condition):

  ```
  fields=elementCollection/Element/attributes/(Geography contains or,Asia,Europe).
  ```

- For parent folder attribute, prefix the filtering condition with `r`, to set recursive rule.

  For example, the following example filters all elements where the parent folder is `System` or any subfolder of `System`:

  ```
  fields=elementCollection/Element/attributes/(Parent Folder is r,System).
  ```

- **paging**, **pageSize**, **pageNo**

Use the `paging`, `pageSize`, and `pageNo` parameters to retrieve the collection of elements that are paginated. When pagination is enabled, the JSON shows an additional link to the next page.

Types: `paging`: Boolean, `pageSize`: Integer, `pageNo`: Integer

For example:

```
"paging": {
"next":
"https://fpserver.com:9443/fp/resources/workspaces/125/modules/19/elements/?view=500&pageNo=1&paging=true&pageSize=2",
"pageNo": "0"
}
```

```
paging=true&pageSize=2&pageNo=1
```

Note The user can also supply `pageSize` or `pageNo` as a parameter.

- **includeHistoryOfAttributes**, **start**, **end**

  Set the `includeHistoryOfAttribute` parameter to `true` and specify the `start` and `end` date parameters to retrieve the history of the attributes. The following formats are supported for the date parameters.

  ```
  yyyy-MM-dd'T'HH:mm:ss
  yyyy-MM-dd'T'HH:mm
  yyyy-MM-dd
  yyyy-MM-dd HH:mm:ss
  yyyy-MM-dd HH:mm
  ```

  Types: `includeHistoryOfAttributes`: Boolean, `start`: Date or Date Time, `end`: Date or Date time

  For example:

  ```
  includeHistoryOfAttributes=true&start=2011-03-22T12:00:00+01:00&end=2011-06- 22T12:00:00+01:00
  ```

  ```
  includeHistoryOfAttributes=true&start=2011-03-22T12:00+GMT+01:00&end=2011-06-22T12:00+GMT+01:00
  ```

- **genericTimeGrid**

  Set the `genericTimeGrid` parameter to `true` to retrieve the time grid attribute sheets in a generic format. This can be used for reporting by using Rational Insight.

  Note The user can also supply `genericTimeGrid` as a parameter.

  Type: Boolean

- **optimize**

  Set the `optimize` parameter to `true` to optimize the content. If this is set to `true`, the response does not include rich text and formula information of the attributes. Also alternate URLs to the element are not provided in the response.

  Type: Boolean

- **includeLinkTable**

  Set the `includeLinkTable` parameter to `true` to include attributes from the link targets in the response. The view should have a link table defined for the Link or LinkList attributes. All visible attributes shown in the link table set up will be part of the Link or LinkList attribute response.

  Type: Boolean

- **sortID**

  Use the `sortID` parameter to sort the elements in the collection based on an attribute in the view. Provide the id of the attribute as the value for the parameter.

  Type: Integer

  For example, `sortID=100`.

- **descending**

  Set the `descending` parameter to `true` to sort the elements in the collection in descending order. If this parameter is not specified, sorting is done in ascending order.

  Type: Boolean

POST method

Use the POST method to create a new element in a module.

The following example uses the POST method to create new elements where the user must have administrator permission for creating elements in the module.

URL:
`https://fpserver.com:9443/fp/resources/workspaces/3/modules/1/elements/.json`

Accept header: `application/json`

Request body:

```
{"attributes":[{"Title":{"textValue":"Elem005"}},{"Choice":{"value":"Implemented"}},{"Date":{"value":"2021-11-22T00:00:00+05:30"}},{"Integer":{"value":"50"}},{"Link":{"value":"https://fpserver.com:9443/fp/resources/workspaces/116/modules/13/elements/40"}},{"List":{"linklist":[{"value":"https://fpserver.com:9443/fp/resources/workspaces/116/modules/13/elements/99?view=23"},{"value":"https://fpserver.com:9443/fp/resources/workspaces/116/modules/13/elements/102?view=23"}]}}]}
```

Supported parameters for the POST method

The following parameters can be used with the POST method:

- **optimize**

  Set the `optimize` parameter to true to optimize element creation.

  Type: Boolean

Using the POST method with the view parameter

The following example shows the method that allows non-administrators to create new elements by using the view parameter.

URL:

`https://fpserver.com:9443/fp/resources/workspaces/3/modules/1/elements/.json?view=200&optimize=true`

Accept header: `application/json`

Request body:

```
{"attributes":[{"Title":{"textValue":"Elem005"}},{"Choice":{"value":"Implemented"}},{"Date":{"value":"2021-11-22T00:00:00+05:30"}},{"Integer":{"value":"50"}},{"Link":{"value":"https://fpserver.com:9443/fp/resources/workspaces/116/modules/13/elements/40"}},{"List":{"linklist":[{"value":"https://fpserver.com:9443/fp/resources/workspaces/116/modules/13/elements/99?view=23"},{"value":"https://fpserver.com:9443/fp/resources/workspaces/116/modules/13/elements/102?view=23"}]}}]}
```

Multipart POST method

The following example uses the POST method to update multiple attributes through one HTTP request. The content type for this request is `multipart/related`. You can use the POST method only to add or delete element links.

URL:

`https://fpserver.com:9443/fp/resources/workspaces/5/modules/1/elements/.json`

Accept header: `multipart/related`

Request body:

```
{"attributes":[{"elementId":"114","attributeId":"272","addlink":["https://fpserver.com:9443/fp/resources/workspaces/116/modules/13/elements/102","https://fpserver.com:9443/fp/resources/workspaces/116/modules/13/elements/103"],"deletelink":["https://fpserver.com:9443/fp/resources/workspaces/116/modules/13/elements/32"]}]}
```

PUT method

This example shows how to use the PUT method to update multiple attributes through one HTTP request. The content type for this request is `multipart/related`.

URL:

`https://fpserver.com:9443/fp/resources/workspaces/5/modules/1/elements/.json`

Accept header: `multipart/related`

Request body:

```
{"attributes":[{"elementId":99,"attributeId":257,"textValue":"ele-99"},
{"elementId":100,"attributeId":257,"textValue":"ele-100"}]}
```

Elements

An element contains all the attribute values of the element.

GET method

Use this method to retrieve the element details. You can use the View parameter to filter the attributes.

URL:
`https://fpserver.com:9443/fp/resources/workspaces/2/modules/1/elements/001.json`

Accept header: `application/json`

Response body:


```
{"elementId":25,"alias":"b6ea2199-d243-4d4c-875f-
0478b962b121","attributes":[{"Description":{"textValue":"","formattedTextValue":"","alias":"1c
16e493-4a2e-4716-ab3d-
8f8b13bcddd6","id":"3","type":"Text","editLink":"https://fpserver.com:9443/fp/resources/worksp
aces/49/modules/1/elements/25/attributes/3","xhtmlTextValue":"<div></div>","writable":"true"}}
,{"Title":{"textValue":"Infrastructure","formattedTextValue":"Infrastructure","alias":"111ab96
b-35ac-4c78-a67f-
acb46eafdc32","id":"2","type":"Text","editLink":"https://fpserver.com:9443/fp/resources/worksp
aces/49/modules/1/elements/25/attributes/2","xhtmlTextValue":"<div>Infrastructure</div>","writ
able":"true"}},{"ID":{"alias":"05794d66-45d2-402b-a36b-
17657ec6b378","id":"228","type":"UniqueId","value":"001","writable":"false"}}],"links":[{"rel"
:"alternate","href":"https://fpserver.com:9443/fp/resources/workspaces/49/modules/1/elements/2
5?view=427"},{"href":"https://fpserver.com:9443/fp/workspace/49/element/25"}]}
```


Supported parameters for the GET method

The following parameters can be used with the GET method:

- **view**

  Use the `view` parameter to specify the view ID for retrieving the attributes that are in the view.

  Type: Integer

- **filter**

  The `filter` parameter must be used with the `view` parameter. Use the `filter` parameter to pass the filter ID for retrieving attributes based on the filter criteria.

  Type: Integer

  For example, `view=14&filter=1`.

- **fields**

  Use the `fields` parameter to pass an XPath expression to retrieve specific set of attributes.

  Type: String

  For example, `fields=attributes/(ID|Title|Owner)`.

Note '|' must be encoded as '%7C'

- **includeHistoryOfAttribute**, **start**, **end**

  Set the `includeHistoryOfAttribute` parameter to `true` and specify the `start` and `end` date parameters to retrieve the history of the attributes. The following formats are supported for the date parameters.

  ```
  yyyy-MM-dd'T'HH:mm:ss
  yyyy-MM-dd'T'HH:mm
  yyyy-MM-dd
  yyyy-MM-dd HH:mm:ss
  yyyy-MM-dd HH:mm
  ```

  Types: `includeHistoryOfAttributes`: Boolean, `start`: Date or Date Time, `end`: Date or Date time

  For example:

  ```
  includeHistoryOfAttributes=true&start=2011-03-22T12:00:00+01:00&end=2011-06- 22T12:00:00+01:00
  ```

  ```
  includeHistoryOfAttributes=true&start=2011-03-22T12:00+GMT+01:00&end=2011-06-22T12:00+GMT+01:00
  ```

- **genericTimeGrid**

  Set the `genericTimeGrid` parameter to `true` to retrieve the time grid attribute sheets in a generic format. This can be used for reporting by using Rational Insight.

  Note The user can also supply `genericTimeGrid` as a parameter.

  Type: Boolean

- **optimize**

  Set the `optimize` parameter to `true` to optimize the content. If the `optimize` parameter is set to `true`, the response does not include rich text and formula information of the attributes. Also alternate URLs to the element are not provided in the response.

  Type: Boolean

- **includeLinkTable**

  Set the `includeLinkTable` parameter to `true` to include attributes from the link targets in the response. The view should have a link table defined for the Link or LinkList attributes. All visible attributes that are shown in the link table set up will be part of the Link or LinkList attribute response.

  Type: Boolean

Attributes

The representation of an attribute contains the value of the attribute. Certain attributes have multiple collection of values and each value has attribute URIs. For example: TextList and File. The representation of the single attribute value also includes the edit links for each text entry in the list. You can change the attribute values of elements in the following ways:

- To update the attribute types CheckBox, Choice, Date, Float, Integer, Link, LinkList, Matrix, Multichoice, Text, Time Grid, URL, and UniqueId, use a PUT operation.
- To add, update, or delete a text entry, link entry or a file by using the TextList, LinkList and File attributes, use a POST operation, a PUT operation, or a DELETE operation respectively.
- To increment a Version attribute to the next major version, use a POST operation.

The JSON format to update an attribute value is different from the JSON representation of the value. For example, a Text attribute might have both a text value and an expression, but when the attribute is updated, it can be given either a new text value or an expression.

GET method

Use the GET method to retrieve the attribute details. For example:

URL:
`https://fpserver.com:9443/fp/resources/workspaces/2/modules/1/elements/15/attributes/220.json`

Accept header: `application/json`

Response body:


```
{"List_Text_-_Classic":{"textList":[{"createdDate":"2021-11-
23T17:23:38+05:30","textValue":"Test","formattedTextValue":"Test","author":{"title":"Admin[Sac
hu]"},"link":"https://fpserver.com:9443/fp/resources/workspaces/116/modules/13/elements/114/at
tributes/284/entries/1637668418879","id":"1637668418879","lastChangedBy":{"title":"Admin[Sachu
]"},"lastChangedDate":"2021-11-
23T17:23:38+05:30","xhtmlTextValue":"<div>Test</div>"},{"createdDate":"2021-11-
23T17:23:57+05:30","textValue":"Test In progress","formattedTextValue":"Test In
progress","author":{"title":"Admin[Sachu]"},"link":"https://fpserver.com:9443/fp/resources/wor
kspaces/116/modules/13/elements/114/attributes/284/entries/1637668437270","id":"1637668437270"
,"lastChangedBy":{"title":"Admin[Sachu]"},"lastChangedDate":"2021-11-
23T17:23:57+05:30","xhtmlTextValue":"<div>Test In progress</div>"},{"createdDate":"2021-11-
23T17:24:07+05:30","textValue":"Test Compeleted","formattedTextValue":"<P>Test
Compeleted<BR></P>","author":{"title":"Admin[Sachu]"},"link":"https://fpserver.com:9443/fp/res
ources/workspaces/116/modules/13/elements/114/attributes/284/entries/1637668447669","id":"1637
668447669","lastChangedBy":{"title":"Admin[Sachu]"},"lastChangedDate":"2021-11-
23T17:24:07+05:30","xhtmlTextValue":"<div><p>Test Compeleted\n</p></div>"}],"alias":"9a716201-
f162-4621-8895-
590215a95181","id":"284","type":"List","editLink":"https://fpserver.com:9443/fp/resources/work
spaces/116/modules/13/elements/114/attributes/284","writable":"true"}}
```


Supported parameters for the GET method

The following parameters can be used with the GET method:

- **view**

Use this parameter to specify the view ID for retrieving the attributes that are in the view.

- **filter**

  This must be used with the view parameter. Use this parameter to pass the filter ID for retrieving attributes based on the filter criteria.

  Type: Integer

  For example, `view=14&filter=1`.

- **includeHistoryOfAttribute**, **start**, **end**

  Set the `includeHistoryOfAttribute` parameter to `true` and specify the start and the end dates to retrieve the history of the attributes. The following formats are supported.

  ```
  yyyy-MM-dd'T'HH:mm:ss
  yyyy-MM-dd'T'HH:mm
  yyyy-MM-dd
  yyyy-MM-dd HH:mm:ss
  yyyy-MM-dd HH:mm
  ```

  Types: `includeHistoryOfAttributes`: Boolean, `start`: Date or Date Time, `end`: Date or Date time

  For example:

  ```
  includeHistoryOfAttributes=true&start=2011-03-22T12:00:00+01:00&end=2011-06- 22T12:00:00+01:00
  ```

  ```
  includeHistoryOfAttributes=true&start=2011-03-22T12:00+GMT+01:00&end=2011-06-22T12:00+GMT+01:00
  ```

- **genericTimeGrid**

  Set the parameter to `true` for retrieving the time grid attribute sheets in a generic format. This can be used for reporting using Rational Insight.

  Type: Boolean

POST method

Use this method to add a new entry to the attribute collection. The following examples show how to use the POST method to add new values to an attribute TextList collection.

Note The user must have administrator permission for adding values to the attribute collection.

- Adding new values to a TextList attribute

  URL:
  `https://fpserver.com:9443/fp/resources/workspaces/2/modules/1/elements/53/attributes/237.json`

  Accept header: `application/json`

Request body: `{"textValue":"test 2"}`

- Adding new values to a File attribute

  URL:
  `https://fpserver.com:9443/fp/resources/workspaces/2/modules/1/elements/53/attributes/228.json`

  Accept header: `application/json`

  Request body:
  `{"fileName":"a.txt","contentType":"plain/text","fileData":"YWJj"}`

- Adding new values to a LinkList attribute and deleting a link

  URL:
  `https://fpserver.com:9443/fp/resources/workspaces/2/modules/1/elements/53/attributes/236.json`

  Accept header: `application/json`

  Request body:

```
{
"linklist":[
{"value" :
"https://fpserver.com:9443/fp/resources/workspaces/44/modules/13/elements/40"},
{"value" :
"https://fpserver.com:9443/fp/resources/workspaces/44/modules/13/elements/41"}
]
}
```

PUT method

The following table gives examples of how to update the specified attribute values by using the PUT method:

Attribute type	Request body
Check Box, Lock	`{"value" : "false"}`
Choice	`{"value" : "Duplicate"}`
Date	`{"value" : "2021-11-24T00:00:00+05:30"}`
Float	`{"value" : "2021"}`

Attribute type	Request body

File	URL example:
	`https://fpserver.com:9443/fp/resources/workspaces/3/modules/13/elements/16/attributes/237/files/01.json`
	
	`{"fileName":"b.txt","contentType":"plain/text","fileData":"YWJj"}`

Link	`{"value" : "https://fpserver.com:9443/fp/resources/workspaces/44/modules/13/elements/40"}`
Integer	`{"value" : "20"}`
List (Link)	`{"linklist":[{"value":"https://fpserver.com:9443/fp/resources/workspaces/116/modules/13/elements/99"},{"value":"https://fpserver.com:9443/fp/resources/workspaces/116/modules/13/elements/102"}]}`

List (Text)	URL example:
	`https://fpserver.com:9443/fp/resources/workspaces/3/modules/13/elements/19/attributes/247/entries/1369815257240.json`
	
	`{"textValue" : "Test 2333"}`

Matrix	`{"row":[{"cell":[{"value":"A"},{"value":"B"}]},{"cell":[{"value":"Row 1"},{"value":"Simple text value"},{"expression":"1+2"}]}]}`
MultiChoice	`{"selected" : ['one', 'two']}`
Text	`{"textValue" : "new text value"}`
Timegrid (Header)	`{"timeGridSetting":{"sheetId":"1","startDate":"2021-12-31","endDate":"2022-03-30"}}`
Timegrid (Data)	`{"cell":[{"sheetId":"1","date":"2021-12-31","columnId":"A","rowId":"1","value":"23"},{"sheetId":"1","date":"2022-01-31","columnId":"B","rowId":"1","value":"25"}]}`
UniqueID	`{"value" : "5555"}`
URL	`{"value" : "https://www.teamblue.unicomsi.com/products/focal-point/"}`

DELETE method

The following table gives examples of how to delete the specified attribute values by using the DELETE method:

Attribute type	Request body
File	For example, to delete the `01.json` file from the file attribute, specify the following URL and use the delete command. `https://fpserver.com:9443/fp/resources/workspaces/3/modules/13/elements/16/attributes/237/files/01.json`
List (Text)	For example, to delete the `1369815257240.json` entry from the list text attribute, specify the following URL and use the delete command. `https://fpserver.com:9443/fp/resources/workspaces/3/modules/13/elements/19/attributes/247/entries/1369815257240.json`

www.unicomsi.com

We welcome feedback on our documentation. Please email us at:
tech.authors@unicomsi.com

www.unicomglobal.com