UNICOM® Global
Putting IT All Together.℠

Focal Point®

Custom Chart Plugin Reference Manual

Release 7.50

UNICOM® Systems, Inc.
A Division of UNICOM Global

Publication information

FPNC-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

Trademarks

Contents

1 **Focal Point Custom Chart Plugin Reference** 5

Introduction 5

Overview Diagram 5

Configuration 6

Examples 6

Steps to create a script for Chart Plugin 6

Step 1 Create a base script 6

Step 2 Create an add-on script (optional step) 7

Step 3 Add CSS, JPEG or JS Files 7

Step 4 Create the Plugin 8

Steps to upload a Custom Chart Plugin 8

Step 1 Go to the Configure Plugin page 8

Step 2 Download the sample Chart Policy 9

Step 3 Unzip the file SampleChartPolicy.zip 9

Step 4 Import the sample workspace (optional step) 10

Step 5 Upload the Plugins 10

Step 6 Configure the plugin on the home page 12

Step 7 Configure the plugin on the Homepage for XSLT plugin 14

Step 8 Configure the plugin on a view page 15

CUSDATA/DAO object 16

CUSDATA API Information 17

getAttribidFromName 17

getAttribNameFromId 17

getAttrNamesFromDAO 17

getBasicFpDataUrl 18

getCurrentPluginId 18

getCustomFpDataUrl 18

getData 19

getFPRestData 20

getPref 20

getPrefData 20

postPrefData 21

resize 21

savePref 21

- transform 22

Data types 22

- ID 22
- Workspace 23
- View 23
- Element 23
- Attribute 23
- AttributeValue 23
- AttributeSetup 25
- ElementSet 25
- AttributeType 25
- AttributeSetupInfo 26
- ChoiceSetup 26
- ChoiceSetupItem 26
- ViewInfo 26
- HistoryEntry 26

1 Focal Point Custom Chart Plugin Reference

Introduction

Data visualization capabilities that are included in Focal Point help you to make value-based decisions in the field of Portfolio Management and Product Management. You can also use the 'Custom Chart Plugin' capability to develop and add your own unique charting/reporting capabilities as plugins.

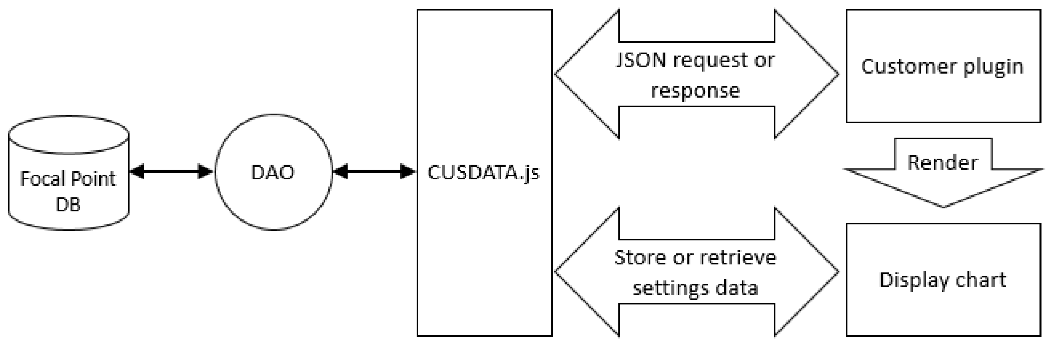
This document describes the API and the Usage guide of 'Custom Chart Plugin' for Focal Point version 7.1.2 and later.

The Focal Point Chart plugin is based on the Focal Point REST services. The purpose is to provide access to Focal Point data and services and facilitate the display of the data in the form of visualization charts and reports on home pages and view pages.

To use the Focal Chart plugin and understand this document, you should be familiar with the Focal Point REST Services, Java Script, and JSON/XML data standards and have basic understanding of the Focal Point data model, usage, and administration.

Overview Diagram

The following diagram gives an overview of the processes involved in configuring a custom chart or report plugin for a view or home page.



The DAO (Data Access Object) contains information about the workspace, views, attributes, and so on. CUSDATA.js a script that contains all utility functions and JS APIs. For more information, see *'CUSDATA/DAO object' on page 16* and *'CUSDATA API Information' on page 17*.

Configuration

As an Admin, you can upload the Custom Chart plugin in the **Applications** > **Plugins** settings and then configure home page settings and view settings.

Charts and reports can be created by using any of the freely available data visualization libraries such as D3, Dimple, Plotly and XSLT.

The Focal Point JavaScript API, which is provided for the development of the Custom Chart plugin, provides access to the following services:

- Retrieve attribute names and types available in a view
- Retrieve data in JSON format using the Focal Point REST API
- Retrieve user settings from Focal Point and save back to database

Examples

Code examples for Custom Chart plugins in Focal Point can be found in the download link provided in **Administration** > **Application** > **Plugin** page.

The examples provided can be used as the basis for custom chart development.

Steps to create a script for Chart Plugin

Step 1 Create a base script

The path to the script has two parts; one is fixed and the other is a relative path to the script file.

The fixed path (same for all plugins) is one of the following paths:

- For homepage plugins: `fpone/plugins/homepage/`
- For view plugins: `fpone/plugins/view/`

The relative path to script (different for each plugin) is in the following form:

```
<DIR PATH>/<scriptfilename>
```

The complete path is the fixed path concatenated with the relative path.

For example:

```
fpone/plugins/view/MYDIR/main
```

The base script file (`main.js`) has the following structure.

```
define("fpone/plugins/view/MYDIR/main",[
     "dojo/ready"
      ], function(ready) {
function main(CUSDATA) {
}
     return main;
});
```

Step 2 Create an add-on script (optional step)

The following step is optional, and is only required if you want to add another script to the base script.

First create a script in AMD format which would look like the following example.

Note For view type plugin, the PATH would be: `fpone/plugins/view/MYDIR/lib`

The Relative PATH to the script is: `MYDIR/lib/addon.js`

```
define("fpone/plugins/view/MYDIR/lib", [], function() {
var Refresh = {};
//User defined functions
Refresh.fullDoc = function(target) {
    //user code
}
    return Refresh;
});
```

Next, to add this script to the base script, modify the base script file (`main.js`) to have the following structure.

```
define("fpone/plugins/homepage/MYDIR/main",[
         "dojo/ready",
         "fpone/plugins/homepage/MYDIR/lib/addon"
         ], function(ready, Refresh) {

function main(CUSDATA) {
   //Call your methods of addon
   //For eg
   Refresh.fullDoc();
}
   return main;
});
```

Step 3 Add CSS, JPEG or JS Files

To add CSS, JPEG or JS Files, create a directory structure to keep these files.

For example:

```
MYDIR/LIB/
MYDIR/CSS/main.css
MYDIR/IMG/clip.png
MYDIR/JS/chart.js
```

In the HTML files just replace the `pluginBasePath` available with the CUSDATA object.

If you are using XSL files, use the following method:

Use the following replacement text:.

For namespace URL use `@@FPNAMESPACE@@`

For example:

```
xmlns:ns="@@FPNAMESPACE@@"
```

For path to custom script files use @@BASEPATH@@

For example:

```
<link rel="stylesheet" type="text/css" href="@@BASEPATH@@XSLT/css/util.css"
/>

<script type="text/javascript"
src=""@@BASEPATH@@XSLT/lib/jquery.js"></script>
```

Note Refer to the sample XSL file in the XSLT zip folder (`demoxslt.xsl`).

Note If there is a single CSS file, then name the CSS file the same as the base script file and it will be automatically picked up. For example, if the base filename is `main.js` then name the CSS filename as `main.css`. In this case the CSS file should be present in the same directory as that of the base script file.

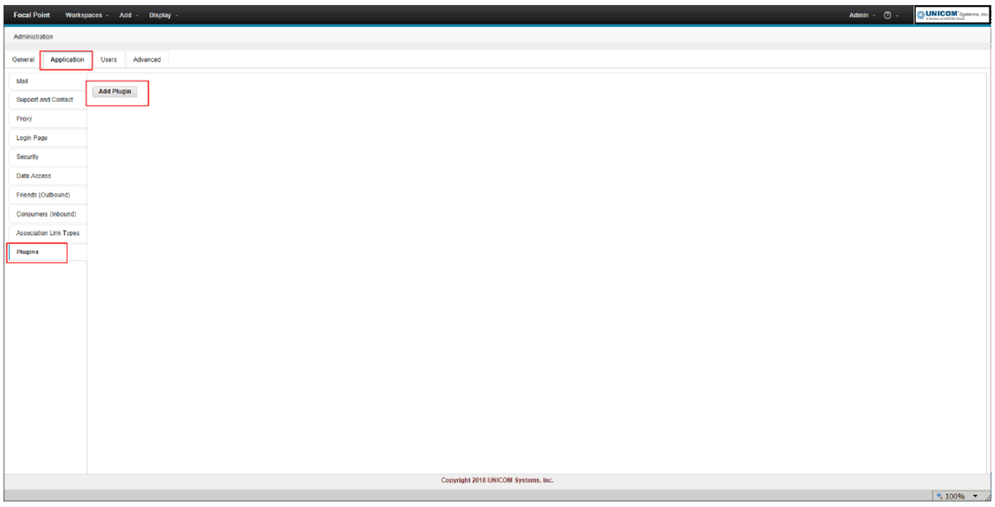
Step 4 Create the Plugin

The final step is to create a zip file of the directory `MYDIR` and upload it as a plugin.

Steps to upload a Custom Chart Plugin

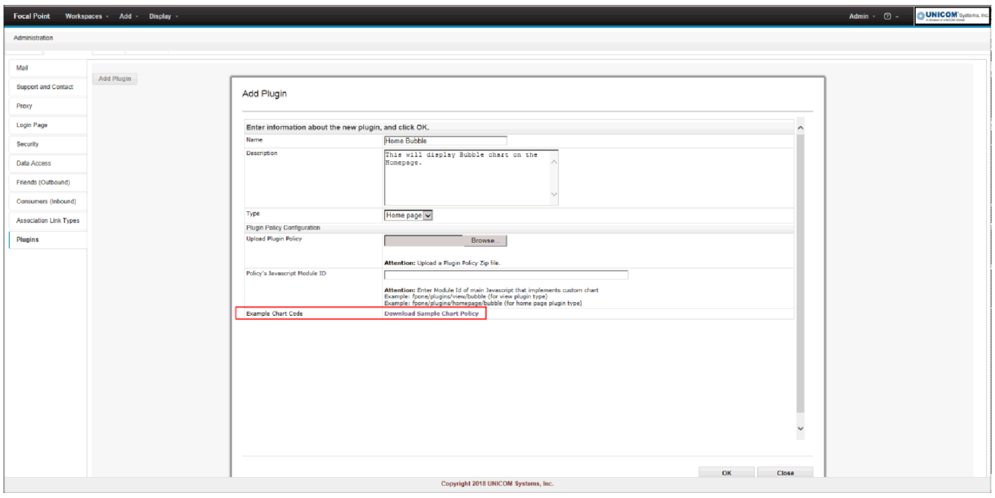
Step 1 Go to the Configure Plugin page

Login as Admin and go to **Administration** > **Application Tab** > **Plugins** and then click **Add Plugin**.



Step 2 Download the sample Chart Policy

When the Plugin form is displayed, click on the link **Download Sample Chart Policy**.



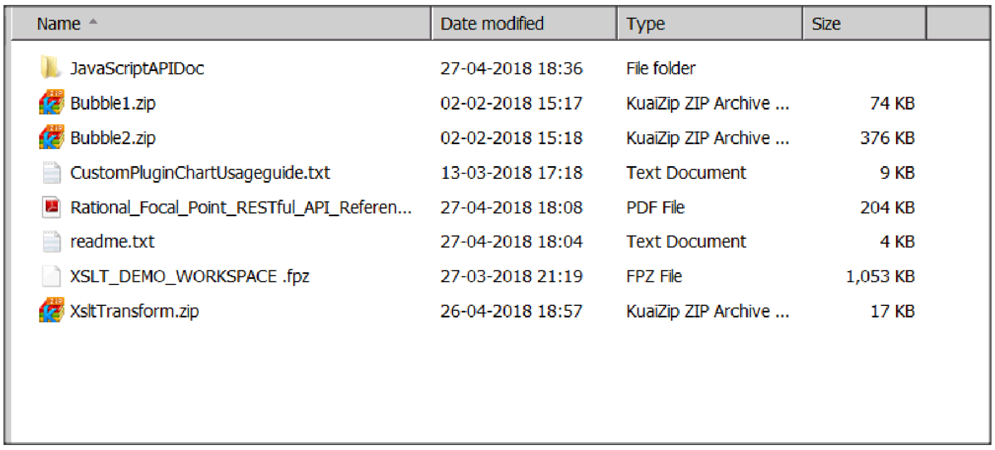
Add Plugin

Enter information about the new plugin, and click OK.

Name	Home Bubble
Description	This will display Bubble chart on the Homepage.
Type	Home page
Plugin Policy Configuration	
Upload Plugin Policy	Browse... **Attention:** Upload a Plugin Policy Zip file.
Policy's Javascript Module ID	**Attention:** Enter Module Id of main Javascript that implements custom chart Example: fpone/plugins/view/bubble (for view plugin type) Example: fpone/plugins/homepage/bubble (for home page plugin type)
Example Chart Code	Download Sample Chart Policy

Step 3 Unzip the file SampleChartPolicy.zip

Unzip the file `SampleChartPolicy.zip` to display the contents.



Name	Date modified	Type	Size
JavaScriptAPIDoc	27-04-2018 18:36	File folder	
Bubble1.zip	02-02-2018 15:17	KuaiZip ZIP Archive ...	74 KB
Bubble2.zip	02-02-2018 15:18	KuaiZip ZIP Archive ...	376 KB
CustomPluginChartUsageguide.txt	13-03-2018 17:18	Text Document	9 KB
Rational_Focal_Point_RESTful_API_Referen...	27-04-2018 18:08	PDF File	204 KB
readme.txt	27-04-2018 18:04	Text Document	4 KB
XSLT_DEMO_WORKSPACE .fpz	27-03-2018 21:19	FPZ File	1,053 KB
XsltTransform.zip	26-04-2018 18:57	KuaiZip ZIP Archive ...	17 KB

Step 4 Import the sample workspace (optional step)

Note This step is optional and is needed only if you want to deploy the XSLTdemo plugin.

Login as Admin and import the sample workspace `XSLT_DEMO_WORKSPACE.fpz` file.

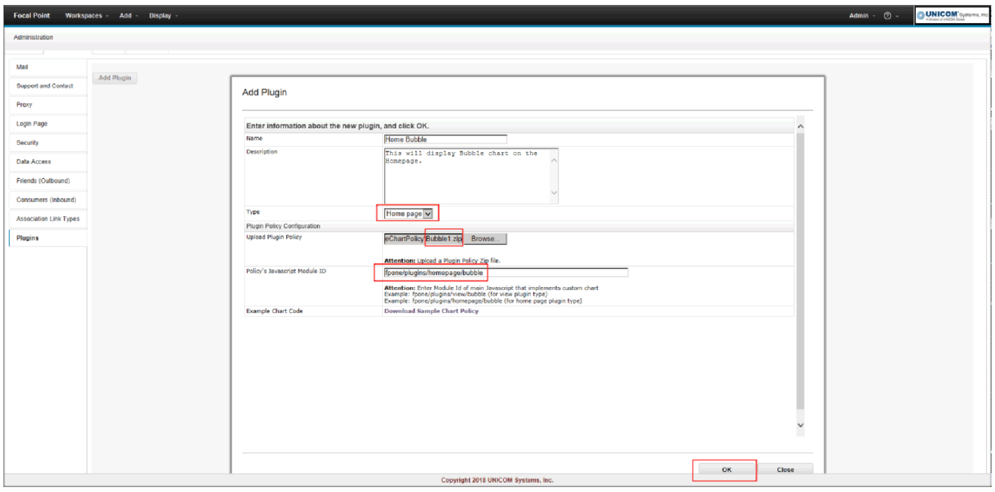
Step 5 Upload the Plugins

Now proceed to fill the **Add Plugin** Form with all the necessary plugin information as shown in the following steps:

Upload Plugin 1

To display a bubble chart on the home page, select `Bubble1.zip` file against **Upload Plugin Policy**.

Provide 'Policy's Javascript Module ID' as `fpone/plugins/homepage/bubble`. Click on the **OK** button when the setup is complete.

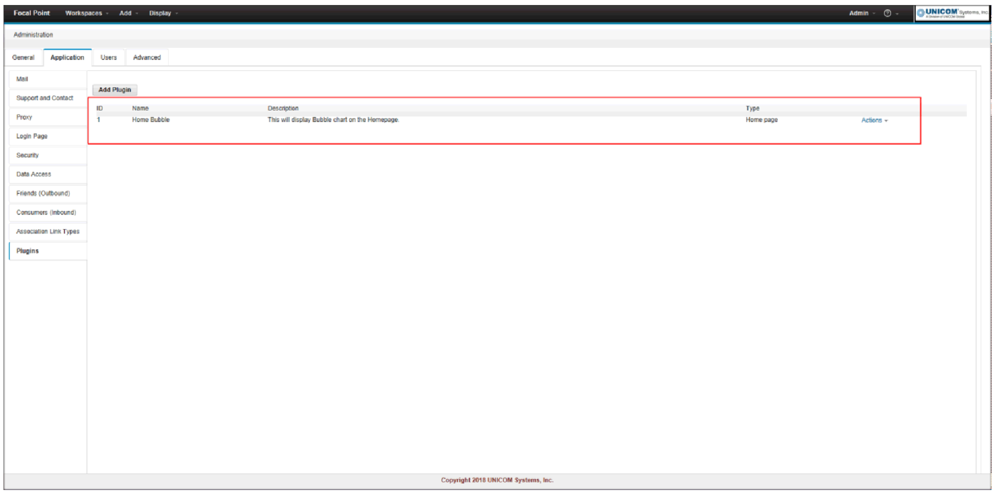


Add Plugin

Enter information about the new plugin, and click OK.

Name	Home Bubble
Description	This will display Bubble chart on the Homepage.
Type	Home page
Plugin Policy Configuration	
Upload Plugin Policy	eChartPolicy\Bubble1.zip Browse... **Attention:** Upload a Plugin Policy Zip file.
Policy's Javascript Module ID	fpone/plugins/homepage/bubble **Attention:** Enter Module Id of main Javascript that implements custom chart Example: fpone/plugins/view/bubble (for view plugin type) Example: fpone/plugins/homepage/bubble (for home page plugin type)
Example Chart Code	Download Sample Chart Policy

After the upload process is successful, the "Plugins" page displays an entry as shown in the following screenshot:

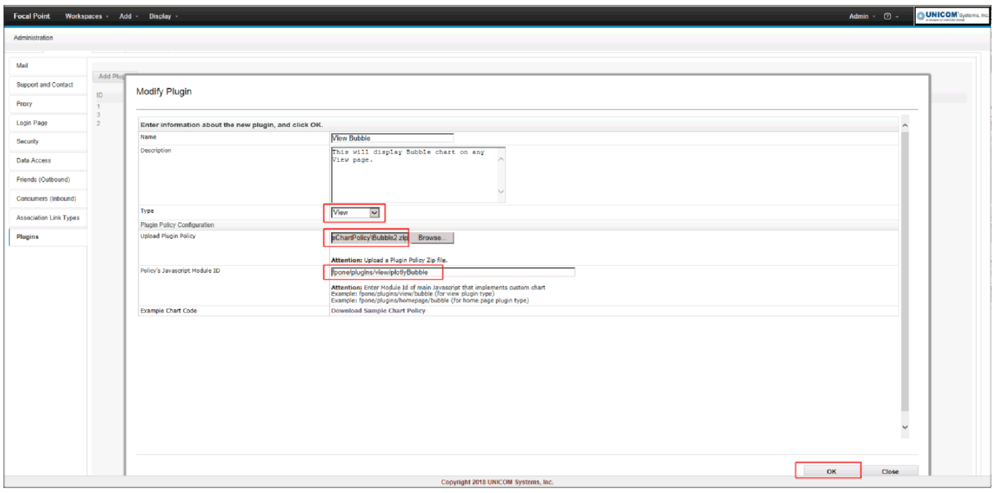


ID	Name	Description	Type	
1	Home Bubble	This will display Bubble chart on the Homepage.	Home page	Actions

Upload Plugin 2 (optional step)

To display a bubble chart on a view page, select `Bubble2.zip` file against ‘Upload Plugin Policy’. Entry against ‘Policy's Javascript Module ID’ will be `fpone/plugins/view/plotlyBubble`.

Click on the **OK** button, when the setup is complete.



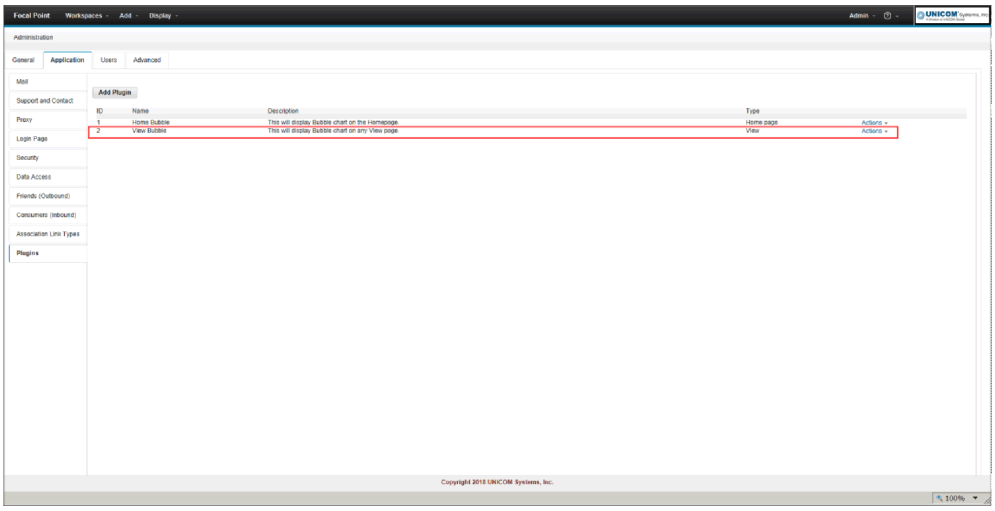
Modify Plugin

Enter information about the new plugin, and click OK.

Name	View Bubble
Description	This will display Bubble chart on any View page.
Type	View
Plugin Policy Configuration	
Upload Plugin Policy	eChartPolicy\Bubble2.zip Browse... **Attention:** Upload a Plugin Policy Zip file.
Policy's Javascript Module ID	fpone/plugins/view/plotlyBubble **Attention:** Enter Module Id of main Javascript that implements custom chart Example: fpone/plugins/view/bubble (for view plugin type) Example: fpone/plugins/homepage/bubble (for home page plugin type)
Example Chart Code	Download Sample Chart Policy

Copyright 2018 UNICOM Systems, Inc.

After the upload process is successful, the “Plugins” page displays an entry as shown in the following screenshot:



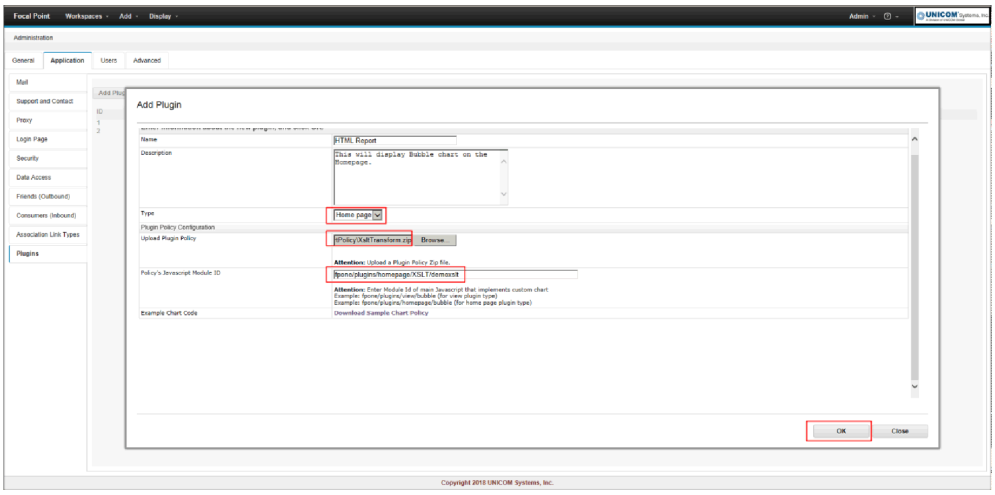
ID	Name	Description	Type	
1	Home Bubble	This will display Bubble chart on the Homepage.	Home page	Actions
2	View Bubble	This will display Bubble chart on any View page.	View	Actions

Copyright 2018 UNICOM Systems, Inc.

Upload Plugin 3 (optional step)

For displaying a HTML report on the home Page, select `XslTransform.zip` file against **Upload Plugin Policy**. Provide the value `fpone/plugins/homepage/XSLT/demoxslt` against ‘Policy's Javascript Module ID’.

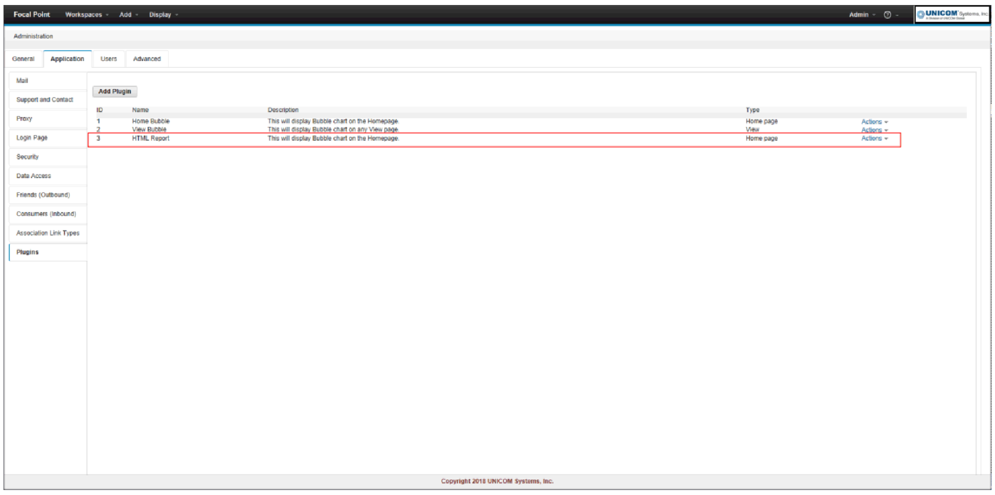
Click on the **OK** button, when the setup is complete.



Add Plugin

Name	HTML Report
Description	This will display Bubble chart on the Homepage.
Type	Home page
Plugin Policy Configuration	
Upload Plugin Policy	tPolicy\XsltTransform.zip **Attention:** Upload a Plugin Policy Zip file.
Policy's Javascript Module ID	fpone/plugins/homepage/XSLT/demoxslt **Attention:** Enter Module Id of main Javascript that implements custom chart Example: fpone/plugins/view/bubble (for view plugin type) Example: fpone/plugins/homepage/bubble (for home page plugin type)
Example Chart Code	Download Sample Chart Policy

After the upload process is successful, the “Plugins” page displays an entry as shown in the following screenshot:

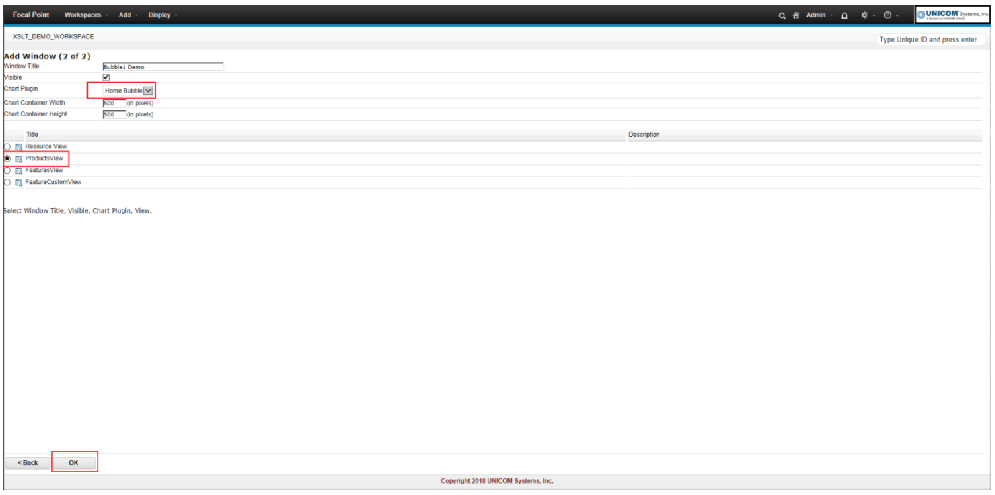


ID	Name	Description	Type	
1	Home Bubble	This will display Bubble chart on the Homepage.	Home page	Actions
2	View Bubble	This will display Bubble chart on any View page.	View	Actions
3	HTML Report	This will display Bubble chart on the Homepage.	Home page	Actions

Step 6 Configure the plugin on the home page

Open **XSLT DEMO WORKSPACE**, go to **Homepage Settings** > **Add window** and click on the **Chart** option. The Add window form will be displayed.

Proceed by filling in the necessary details like Window title, Chart Plugin and View.



XSLT_DEMO_WORKSPACE

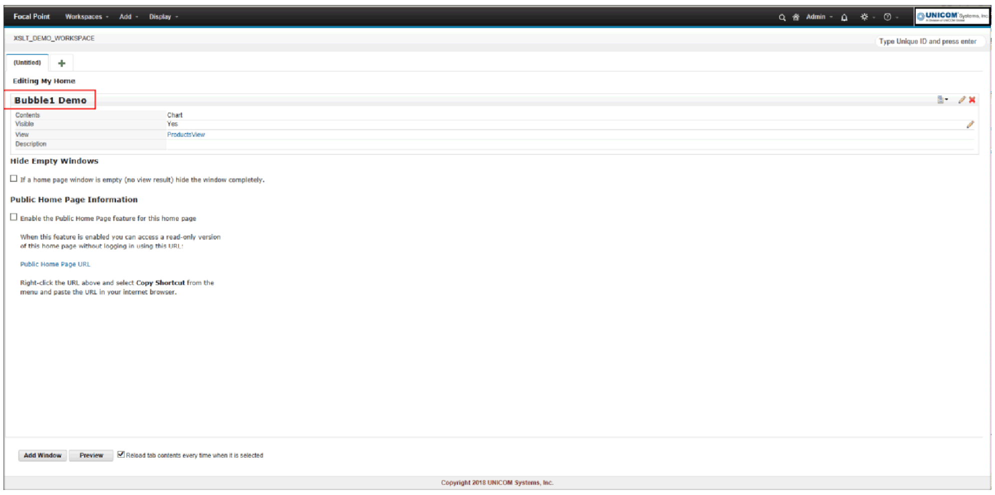
Add Window (2 of 2)

Window Title	Bubble1 Demo
Visible	✓
Chart Plugin	Home Bubble
Chart Container Width	600 (in pixels)
Chart Container Height	500 (in pixels)

	Title	Description
○	Resource View	
●	ProductsView	
○	FeaturesView	
○	FeatureCustomView	

Select Window Title, Visible, Chart Plugin, View.

On clicking the **OK** button, the following page is displayed.



XSLT_DEMO_WORKSPACE

Editing My Home

Bubble1 Demo

Contents	Chart
Visible	Yes
View	ProductsView
Description	

Hide Empty Windows

☐ If a home page window is empty (no view result) hide the window completely.

Public Home Page Information

☐ Enable the Public Home Page feature for this home page

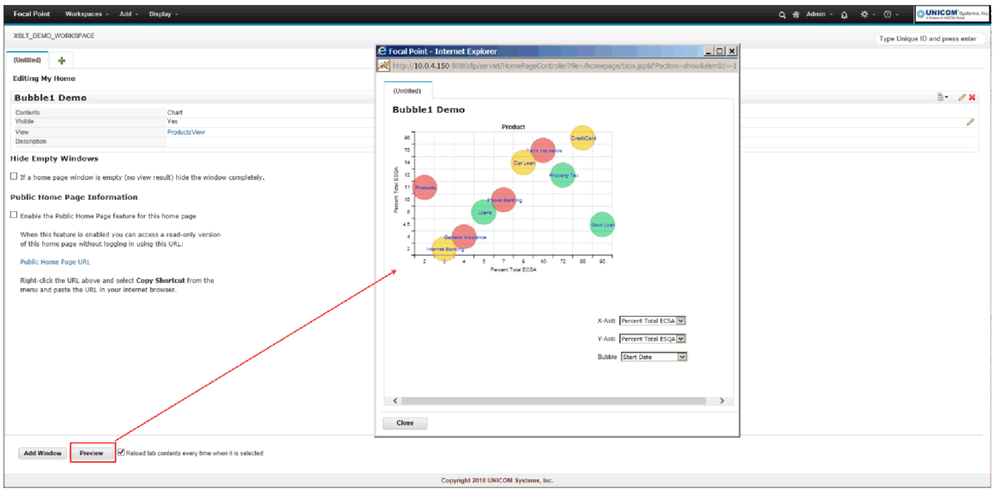
When this feature is enabled you can access a read-only version of this home page without logging in using this URL:

Public Home Page URL

Right-click the URL above and select **Copy Shortcut** from the menu and paste the URL in your internet browser.

✓ Reload tab contents every time when it is selected

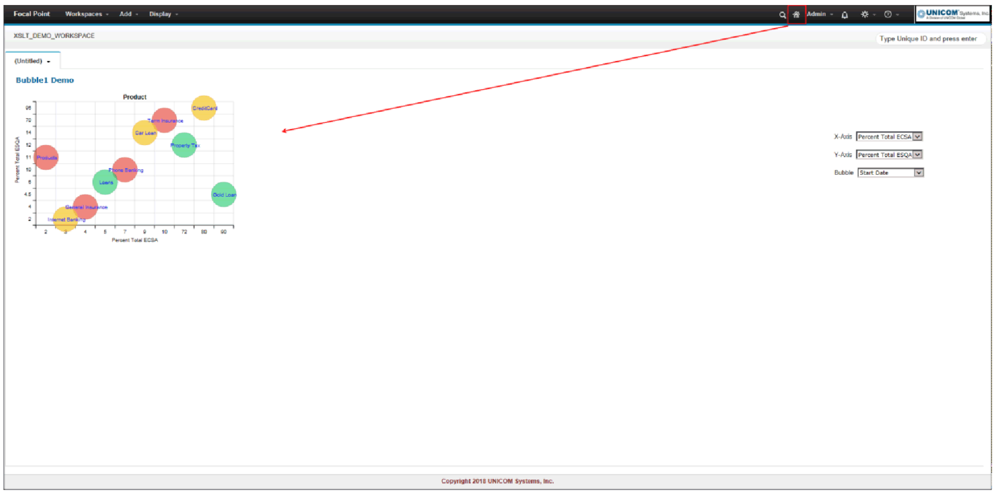
Click on the **Preview** button to see the chart being displayed.



X-Axis	Percent Total ECSA
Y-Axis	Percent Total ESQA
Bubble	Start Date

✓ Reload tab contents every time when it is selected

Click on the Home icon to see the home page.



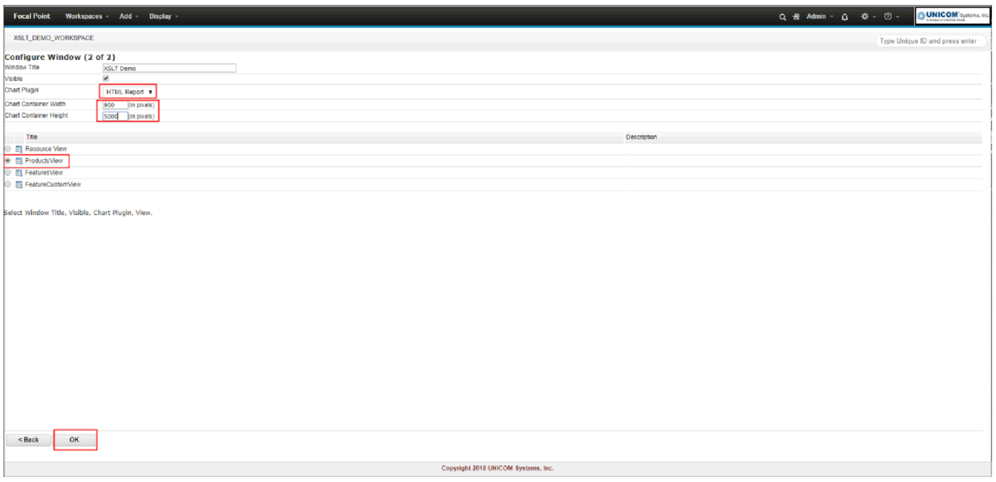
X-Axis	Percent Total ECSA
Y-Axis	Percent Total ESQA
Bubble	Start Date

Change the values on X-axis, Y-Axis and bubble to see the changes being applied to the chart.

Step 7 Configure the plugin on the Homepage for XSLT plugin

Open the **XSLT DEMO WORKSPACE**, go to **Homepage Settings** > **Add window** and click on the **Chart** option. The Add window form will be displayed.

Proceed by filling in the necessary details like Window title, Chart Plugin, and View. Set the height=5000 and width=900.


Focal Point
Workspaces
Add
Display
Admin
XSLT_DEMO_WORKSPACE
Type Unique ID and press enter
< Back
OK
Copyright 2018 UNICOM Systems, Inc.

Configure Window (2 of 2)

Window Title	XSLT Demo
Visible	✓
Chart Plugin	HTML Report
Chart Container Width	900 (in pixels)
Chart Container Height	5000 (in pixels)

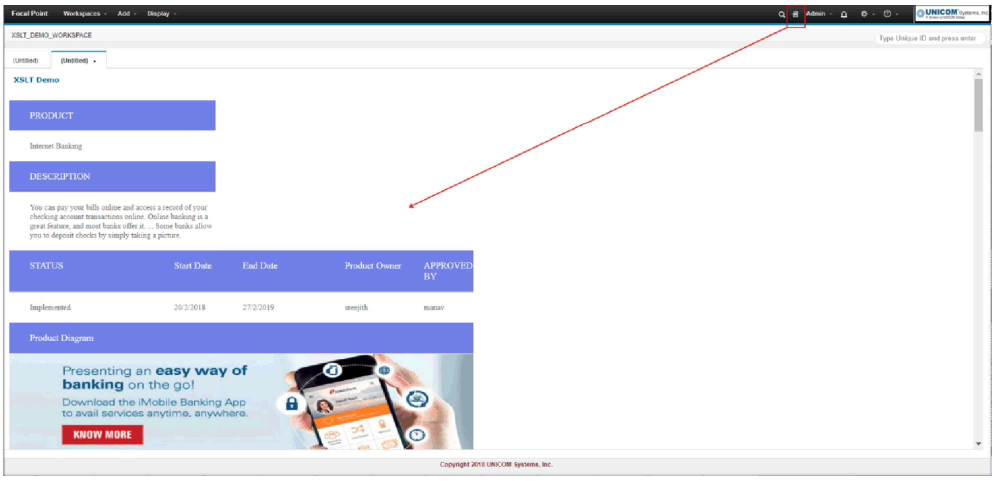
	Title	Description
○	Resource View	
●	ProductsView	
○	FeaturesView	
○	FeatureCustomView	

Select Window Title, Visible, Chart Plugin, View.

Click on the **OK** button.

Click on the **Preview** button to see the chart being displayed.

Click on the Home icon to see the Home Page.


XSLT Demo

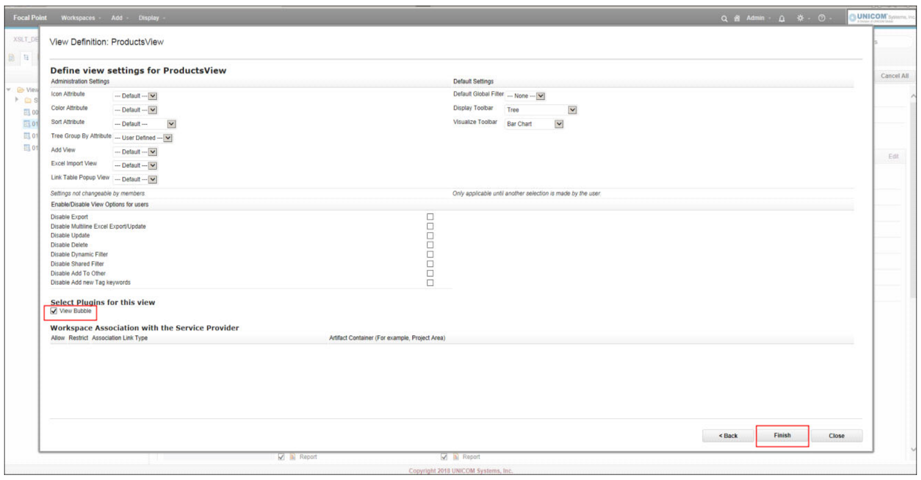
PRODUCT				
Internet Banking				
DESCRIPTION				
You can pay your bills online and access a record of your checking account transactions online. Online banking is a great feature, and most banks offer it. ... Some banks allow you to deposit checks by simply taking a picture.				
STATUS	Start Date	End Date	Product Owner	APPROVED BY
Implemented	20/2/2018	27/2/2019	sreejith	manav

Product Diagram

Focal Point
Workspaces
Add
Display
Admin
XSLT_DEMO_WORKSPACE
(Untitled)
Type Unique ID and press enter
Presenting an easy way of banking on the go!
Download the iMobile Banking App to avail services anytime, anywhere.
KNOW MORE
Copyright 2018 UNICOM Systems, Inc.

Step 8 Configure the plugin on a view page

Go to Configure views, select any view and Edit the view definition and click **Next** twice to reach the Define view settings page. Select **Plugins for this view** by selecting the checkboxes for appropriate plugin and click **Finish** as shown in the following screenshot.


View Definition: ProductsView

Define view settings for ProductsView

Administration Settings		Default Settings	
Icon Attribute	--- Default ---	Default Global Filter	--- None ---
Color Attribute	--- Default ---	Display Toolbar	Tree
Sort Attribute	--- Default ---	Visualize Toolbar	Bar Chart
Tree Group By Attribute	--- User Defined ---		
Add View	--- Default ---		
Excel Import View	--- Default ---		
Link Table Popup View	--- Default ---		

Settings not changeable by members. *Only applicable until another selection is made by the user.*

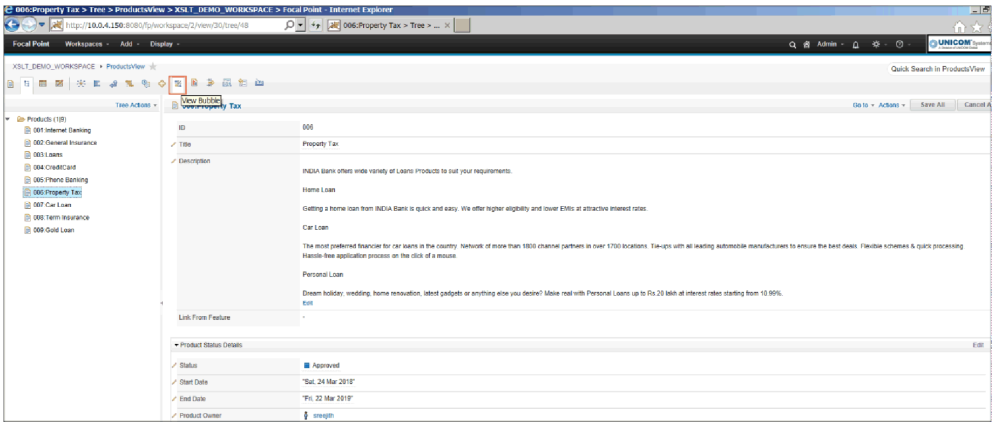
Enable/Disable View Options for users

Disable Export	☐
Disable Multiline Excel Export/Update	☐
Disable Update	☐
Disable Delete	☐
Disable Dynamic Filter	☐
Disable Shared Filter	☐
Disable Add To Other	☐
Disable Add new Tag keywords	☐

Select Plugins for this view
☑ View Bubble

Workspace Association with the Service Provider
Allow Restrict Association Link Type Artifact Container (For example, Project Area)

Now go to Display View and select the view, and click on the icon to display the chart on the view Page, as shown in the following screenshot.


006:Property Tax > Tree > ProductsView > XSLT_DEMO_WORKSPACE > Focal Point - Internet Explorer
http://10.0.4.150:8080/fp/workspace/2/view/30/tree/48

XSLT_DEMO_WORKSPACE • ProductsView

Products (1|9)
- 001 Internet Banking
- 002 General Insurance
- 003 Loans
- 004 CreditCard
- 005 Phone Banking
- 006 Property Tax
- 007 Car Loan
- 008 Term Insurance
- 009 Gold Loan

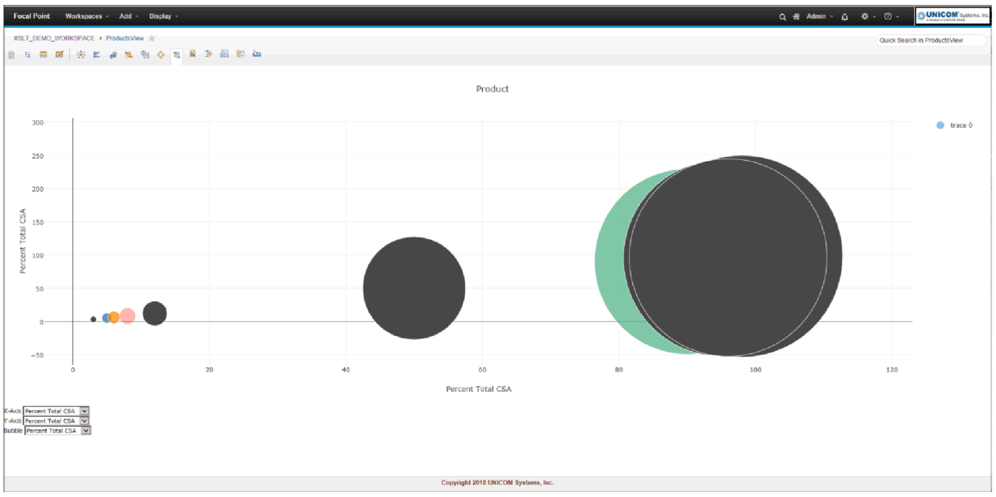
006:Property Tax

ID	006
Title	Property Tax
Description	INDIA Bank offers wide variety of Loans Products to suit your requirements. Home Loan Getting a home loan from INDIA Bank is quick and easy. We offer higher eligibility and lower EMIs at attractive interest rates. Car Loan The most preferred financier for car loans in the country. Network of more than 1800 channel partners in over 1700 locations. Tie-ups with all leading automobile manufacturers to ensure the best deals. Flexible schemes & quick processing. Hassle-free application process on the click of a mouse. Personal Loan Dream holiday, wedding, home renovation, latest gadgets or anything else you desire? Make real with Personal Loans up to Rs.20 lakh at interest rates starting from 10.99%.
Link From Feature	-

Product Status Details

Status	Approved
Start Date	"Sat, 24 Mar 2018"
End Date	"Fri, 22 Mar 2019"
Product Owner	sreejith

On click, the following chart is displayed.



CUSDATA/DAO object

This section provides information about the values present in the CUSDATA/DAO object. Some of the useful information in the CUSDATA object is shown in the following code.

The DAO object in custom script can be accessed by using **JSON Object** > **CUSDATA.dao**.

```
{
"dao":{
   "activeFilterId":-1,
   "moduleNamePlur":"Products",
   "viewId":30,
   "fpresourceUrl":"http://10.0.4.150:8080/fp/resources",
   "ncModuleNamePlur":"Products",
   "moduleName":"Product",
   "treeElement":48,
   "attrsInfo":[],
   "ncModuleName":"Product",
   "moduleId":13,
   "userId":"13",
   "workspaceId":2,
   "divId":"fp-maincontainer"
},
"basicFpDataUrl":"http://10.0.4.150:8080/fp/resources/workspaces/2/modules/
13/elements/?view=30&includeAttributes=true&optimize=true",
"namespaceFPUrl":"http://10.0.4.150:8080/fp/namespace/workspaces/2/modules/
13/views/30/elements",
"attrNames":[],
"attrTypes":[],
"attrRealNames":[],
"attrIds":[],
"pluginBasePath":"../../../../one/js/modules/plugins/view/"
}
```

Note The Attribute Info array contains array length() equal to the number of attributes in the module.

CUSDATA API Information

Open the `JavaScriptAPIDoc` folder unzipped from **`SampleChartPolicy.zip`**. Open the `index.html` File in Browser and all methods could be browsed in detail. Some of the common methods are given in the following sections.

getAttribidFromName

Usage: `getAttribIdFromName(attribName) >` *`{String}`*

Returns the attribute Id for a given Attribute Name.

Parameters:

Name	**Type**	**Description**
`attribName`	String	Attribute name.

Returns:

attribId - Attribute ID

Type: String

getAttribNameFromId

Usage: `getAttribNameFromId(attribId) >` *`{String}`*

Returns the attribute Name for a given Attribute Id.

Parameters:

Name	**Type**	**Description**
`attribId`	String	Attribute ID.

Returns:

attribName - Attribute Name

Type: String

getAttrNamesFromDAO

Usage: `getAttrNamesFromDAO(types) >` *`{Array}`*

Finds the attribute names based upon the types specified. If no types are specified, then all 'integer' and 'float' attributes are retrieved.

Parameters:

Name	**Type**	**Description**
`Types`	Array	Array of type of attributes specified to prepare the attribute list.

Returns:

keys - Array of names of attributes

Type: Array

getBasicFpDataUrl

Usage: `getBasicFpDataUrl() > {String}`

Creates basic URL.

Returns:

Return basic Focal Point DATA URL

Type: String

getCurrentPluginId

Usage: `getCurrentPluginId() > {String}`

Returns the plugin id from DAO object.

Returns:

Plugin id

Type: String

getCustomFpDataUrl

Usage: `getCustomFpDataUrl(ATTR, conditions) > {String}`

Generates a Focal Point REST URL to retrieve data for the attributes from the view in which the plugin is operating. Users can provide a list of attributes for which data needs to be retrieved. It is also possible to provide attribute conditions so that the elements retrieved from the REST URL are filtered based on these conditions.

Refer to the Focal Point REST API for details on how attribute conditions can be specified. This method returns a Focal Point REST URL after the list of attributes and conditions are added to the base URL obtained through the call `getBasicFpDataUrl()`.

Parameters:

Name	**Type**	**Description**
`ATTR`	Array	Array containing the names of the attributes.
`conditions`	Array	Array containing the attribute conditions.

Returns:

Returns custom URL

Type: String

getData

Usage: `getData(ATTR, conditions, callback) > {Void}`

This method returns the collection of attribute values retrieved from the current view by making a REST call to the URL obtained from the method `getCustomFpDataUrl()`. Users can provide a list of attributes for which data needs to be retrieved.

Note Only Integer, Float, Date and Text attributes are supported here.

It is also possible to provide attribute conditions so that the elements retrieved from the REST URL are filtered based on these conditions.

Refer to the Focal Point REST API for details on how attribute conditions can be specified. If the user is looking for the unfiltered response, or is looking for information of attributes other than Integer, Float, Date or Test, then it is recommended to use the method `getFPRestData` which will return the entire response in JSON format.

Parameters:

Name	**Type**	**Description**
`ATTR`	Array	Array containing the names of the attributes.
`conditions`	Array	Array containing the attribute conditions.
`callback`	Function	Callback function with chart data.

Returns:

Type: Void

getFPRestData

Usage: `getFPRestData(URL, callback, outputFormat) > {Void}`

This method returns data in JSON Object or XML format from any Focal Point REST URL.

Note By default the response would be in JSON Format.

User can provide a REST URL as a parameter. To retrieve element and attribute information from the current view in which the plugin is operating, users can pass `@param{basicFpDataUrl}` as defined above. If data needs to be retrieved from any other URL, then users needs to construct the URL accordingly.

Refer to the Focal Point REST API documentation in the help section for details on how a GET URL can be constructed.

Parameters:

Name	Type	Description
`URL`	String	Focal Point REST URL to fetch DATA.
`callback`	Function	Callback function with Focal Point REST response data.
`outputFormat` (optional)	String	Value should be either "json" or "xml". This parameter is optional. If omitted, output would be in JSON Object format.

Returns:

Type: Void

getPref

Usage: `getPref() > {Object}`

Retrieves user preference data from window object.

Returns:

User preference data as a JSON object.

Type: Object

getPrefData

Usage: `getPrefData(callback) > {Void}`

Retrieves the user preference data from the Focal Point Database.

Parameters:

Name	Type	Description
`callback`	Function	Callback functions with user preference data as a parameter.

Returns:

Type: Void

postPrefData

Usage: `postPrefData(callback) >` *`{Void}`*

Saves the user preference data into the Focal Point Database using a HTTP POST call. The data to be saved will be retrieved from the window object using the `getPref()` call. Users should have called `savePref()` to store user preference to the window object before calling this method.

Parameters:

Name	Type	Description
`callback` (optional)	Function	Callback function to handle the success message from the post call. This parameter is optional.

Returns:

Type: Void

resize

Usage: `resize() >` *`{Void}`*

Sets the window dimensions for view and home page.

Returns:

Type: Void

savePref

Usage: `savePref(JsonSettingsData) >` *`{Void}`*

Saves the user preference data into window.fp object for further use.

Parameters:

Name	Type	Description
`JsonSettingsData`	Object	User preference data in JSON format.

Returns:

Type: Void

transform

Usage: `transform(xmlData, userDefinedXslFilePath) > {Void}`

Applies the XSL transformation on the XML data for view or home page.

Parameters:

Name	Type	Description
`xmlData`	Object	Data in XML format.
`userDefinedXslFilePath`	String	Relative XSL File path defined by the user.

Returns:

Type: Void

Data types

This is a description of the data types used by the operations of the Web Service API. They are defined in the WSDL using an inline XSD Schema. When using an object-oriented programming language that can generate code from a WSDL document (for example Microsoft.NET or the Java Apache Axis framework), these data types are typically transferred into classes.

ID

An ID is the combination of a localId and a workspaceId. Focal Point is divided into several workspaces, each with its own identifier. Objects (for example, views, elements, attributes) in each workspace have their own identifier, but it is only unique within that workspace. In order to uniquely identify an object you need both its own identifier (the localId) and the identifier of the workspace (the workspaceId) it is part of. The data type ID is these two identifiers paired together. For example there can be an element with localId 200 in workspace 2 and another element with localId 200 in workspace 3.

Workspace

A Workspace contains information about a workspace in Focal Point. It has an identifier (and integer), a title (the name of the workspace) and a description.

View

A View contains information about a view in Focal Point. It has an ID, a title (the name of the view) and a description.

Element

The data type Element is used to represent an element in Focal Point. It consists of an ID (that uniquely identifies the element) and a list of Attributes.

Attribute

An Attribute is used to maintain the attribute values for an Element. It consists of an “attributeSetupId” which is an ID that references the AttributeSetup that defines the attribute, a “value” which is an AttributeValue that contains the actual value of the attribute and a “displayValue” which is a simple string representation (if possible) of the attribute value.

The displayValue property should be regarded as a read-only property. When updating or creating an element the displayValue of an Attribute will be ignored, it is only the AttributeValue that is used.

If the value of an attribute is empty/null both the AttributeValue property and displayValue will be null.

AttributeValue

AttributeValue is in itself an empty data type. Instead there are a number of data types that extend AttributeValue. In an object-oriented environment this translates to a superclass/subclass structure.

CheckBoxValue

The CheckBoxValue has a Boolean property called selected that tells if the CheckBox attribute is selected or not.

ChoiceValue

The ChoiceValue has an integer that identifies the selected choice. The full list of choices and their names is retrieved from AttributeSetup.

DateValue

The DateValue has an xsd:dateTime that represents the value of the date attribute. For most date attributes in Focal Point only the date and not the time point is stored, but in the DateValue the hour and minute is included as well. In cases where they have not been set in the Focal Point server, the time point 00:00 will be returned from the Web Services.

The time point part will be ignored when setting or updating an attribute value.

FloatValue

The FloatValue has an xsd:double that represents its value.

IntegerValue

The IntegerValue has an xsd:long that represents its value.

LinkValue

The LinkValue is represented by an ID that identifies another element in Focal Point.

ListValue

The ListValue does not contain any values itself. Instead it has a list of values. It can either be a list of LinkValues or a list of TextValues

MultiChoiceValue

The MultiChoiceValue has a list of integers that represents the selected choices. The full list of choices and their names is retrieved from AttributeSetup.

TextValue

The TextValue contains a string with the value of the text attribute. It will be equal to displayValue.

Any formatting used will be ignored.

UniqueIdValue

The UniqueIdValue is represented by a string.

UrlValue

The UrlValue is represented by a string.

VersionValue

The VersionValue is represented by a string.

FileValue

The FileValue contains a list of FileDataValues. The FileDataValue contains the name, content type and length of a file. Each FileDataValue also has a file number, which is an identifier for that particular file within the file attribute. Note that the actual file content is not included when reading a file attribute (for example when using GetElement or GetElements). Retrieving the file content can be done by using GetFileContent. When writing a file (and using AddElement or UpdateElement), the file content is included. If the supplied file number is found in the existing file attribute (on the server) the file is replaced with the supplied data, if the file number is not found a new file is added.

AttributeSetup

The AttributeSetup is the definition of an attribute. An AttributeSetup is always referred to by an Attribute. It contains an ID, a “title” (the name of the attribute), a “description”, a “type” (an AttributeType) and a “mandatoryName”.

The mandatoryName is a string that is used only for attributes that are mandatory (for example title, last changed by). Since it is possible to change the name for any attribute in Focal Point, mandatoryName is a name that does not change. For example the mandatoryName of the description attribute will always be “Description” even if the title has been changed to something else.

Another way to determine which attribute is the title, description, or prefix attribute is by reading the AttributeSetupInfo in a ViewInfo or ElementSet.

ElementSet

An ElementSet is a container of Elements and some information about the view they where retrieved from. It has a list of Elements, a list of AttributeSetups reflecting the attribute setups for the view that the elements were retrieved from and an AttributeSetupInfo.

AttributeType

The AttributeType data type is used by AttributeSetup to tell which type attribute values based on that attribute setup will be. AttributeSetup is an enumeration and can be any of the following values: Text, Link, Date, CheckBox, List, Choice, Float, Integer, MultiChoice, Url, UniqueId, or Version.

Example: If an AttributeSetup has the AttributeType Date any Attribute that refers to the AttributeSetup will have a DateValue value.

AttributeSetupInfo

The AttribteSetupInfo data type contains information about a set of AttributeSetup retrieved from a view. It contains three IDs that each point to an AttributeSetup. This is used to tell which attribute is the title, description, and prefix attribute. It is also possible to determine this by reading the mandatoryName property of an AttributeSetup (see the description of AttributeSetup).

ChoiceSetup

The ChoiceSetup data type is an extension of the AttributeSetup data type. It includes everything an AttributeSetup has got and adds a list of ChoiceSetupItems. The ChoiceSetup is used when the AttributeType in AttributeSetup is Choice. The ChoiceSetupItems contain information about all available choices for a choice attribute.

ChoiceSetupItem

The ChoiceSetupItem data type contains information about a choice for a choice attribute. It has an integer which is an identifier for this choice and a “title”. The id is used to match ChoiceSetupItems and ChoiceValues.

ViewInfo

The ViewInfo data type contains a list of AttributeSetups and an AttributeInfo. That can be useful if you want to know which attributes that are visible in a view without first retrieving an ElementSet, for example if you want to create a new Element.

HistoryEntry

A HistoryEntry describes a previous change of an attribute. It consists of four parts: “username” the full name of the user who made the change, “date” the date of the change, “elementId” the ID of the element that was changed and “attribute” the changed attribute with the value it had at the time of the change.