Content Services Toolkit

Updated: 12/9/2019

Location of Content\_Services – 19.0.3.twx

<https://github.com/icp4a/content-toolkit>

Import the toolkit into BAStudio

Create your application and reference the Content Services toolkit.

Add the Content\_Services toolkit to your application. Find “Toolkits” in left panel and click “+” icon to add the toolkit.

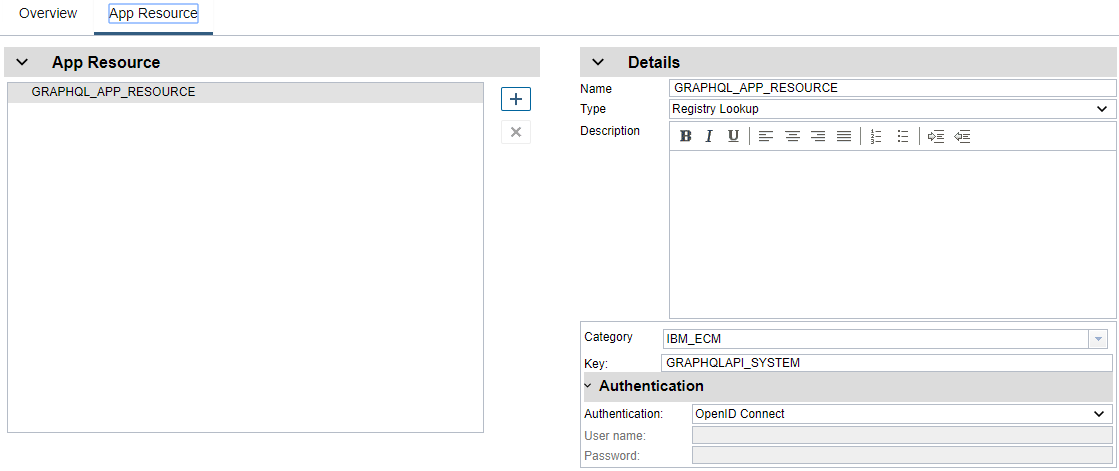
Open your application’s “App Project Settings” view where you define the location of the graphql server

Select the “App Resource” tab, set:

Name = GRAPHQL\_APP\_RESOURCE

Type = Registry Lookup

and enter the following:



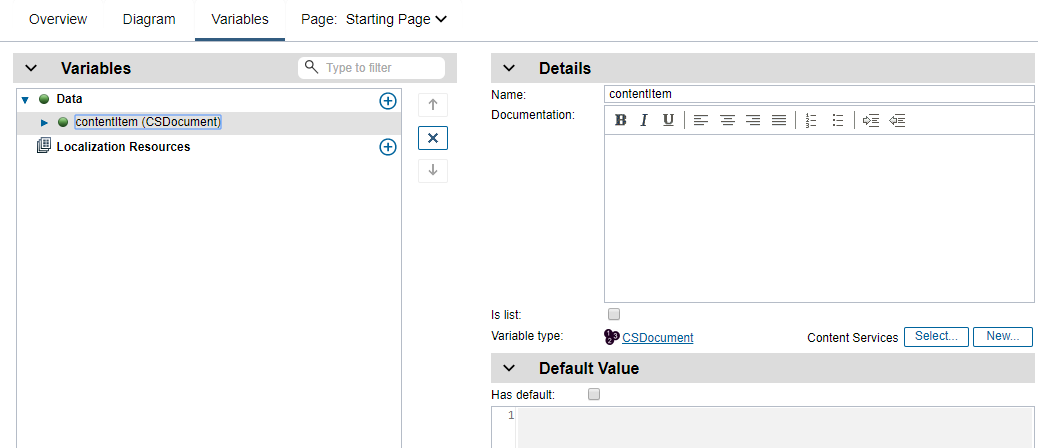
Category: IBM\_ECM

Key: GRAPHQLAPI\_SYSTEM

Authentication: OpenID Connect

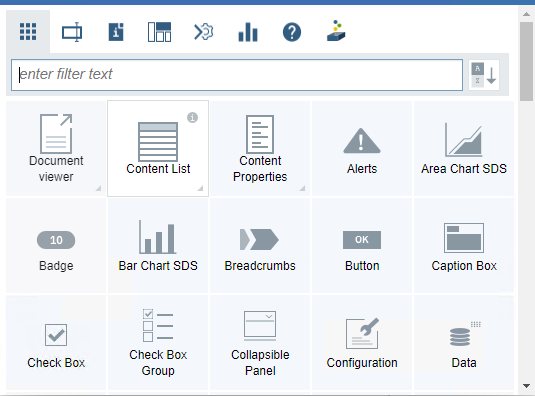
Next, open your application’s “Variables” tab

Create the “contentItem” variable (Is List = false, variable type” CSDocument):

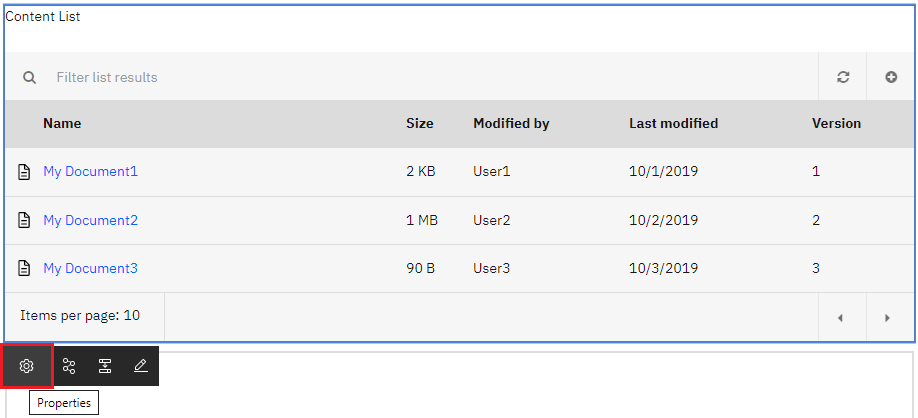


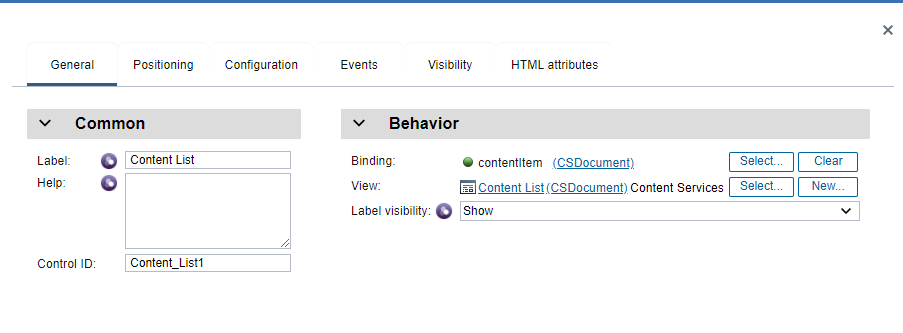
# Content List - Configuration

Open the component selection dialog and select the “Content List” component. It’s in the “Layout” category.



Open the properties dialog (left click on component):





Select the “Configuration” tab

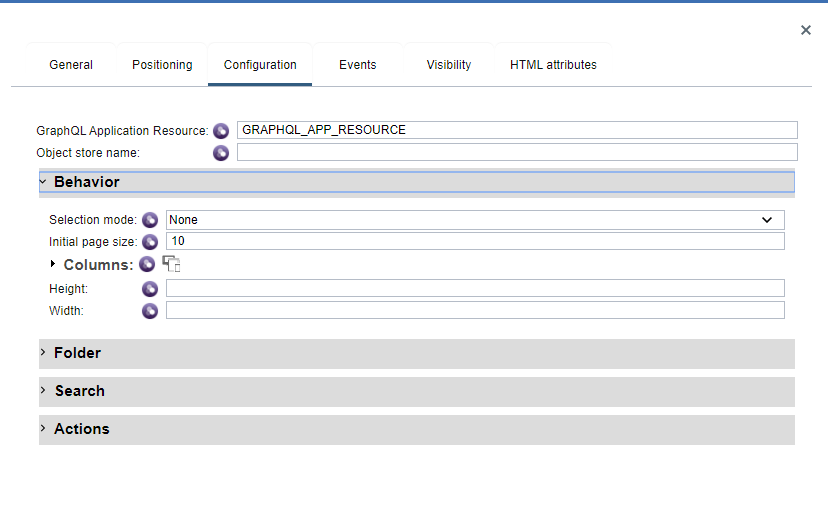
In the GraphQL Application Resource field, enter the graphql app resource name you created above.

e.g. “GRAPHQL\_APP\_RESOURCE”

In the Object store name field, enter the symbolic name of your CE repository (get this from ACCE).

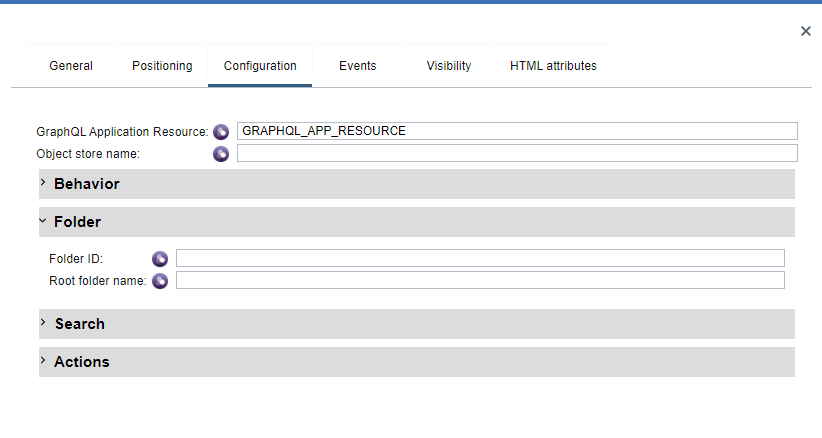
Open the Behavior section.

You can optionally enter a new paging size for the list view. The default is 20 items (if not specified).



If you want the list to display folder contents, open the Folder section and enter the id (guid) of the folder you would like to display. Specifying a folder path instead is also supported (e.g. //My Folder Path).

You can optionally enter the root folder name to display (in case your folder does not have a user-friendly name).



If you would like to display search results, open the Search section and enter the following settings (settings listed with ‘\*’ are required):

**Search Folder ID**: Folder ID (Guid) retrieved from ACCE (or ICN)

**Search only this folder**: Specifies whether to include search sub-folders

**\*Search property**: Symbolic name of property you would like to search on (from ACCE)

**\*Operator**: Supported operator per property type and cardinality, such as Starts With, Ends With, Like, Not Like, =, <, >, <>, >=, <=, etc.

**Default value**: Value to search for. If no default value is entered, the list is initially displayed empty and user must type in search field to initiate a query

**\*Search property type**: The property data type.

**Hide search field**: Can hide the search filed at runtime. The editable flag can be used to lock down the query parameters if you don’t want the user to change them.

**Text Options**: Specify a text search option (CBR), the default is not enabled.

We are still working through some “issues” with properties that have choice lists and multi-value properties.

If the property has a choice list and the display values are different than the actual values, you must type in the actual value. There currently isn’t any mechanism to resolve the display name to the actual value.

Multi-value property support is still being worked on and not fully functional at this point.

There are several special operators that are currently supported:

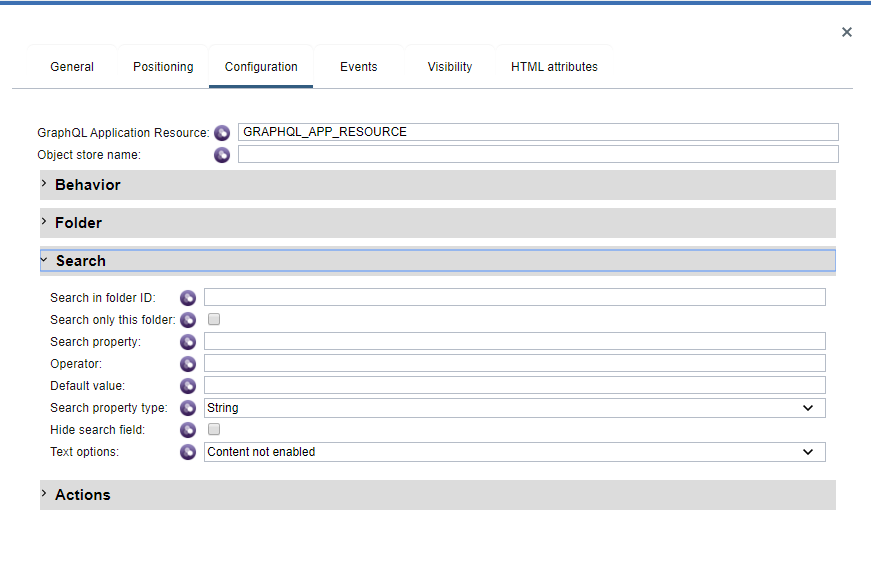
“Include All” – Multi value properties only

“In” – Single value properties with a choice list, this is the “Include Any” operator

“Intersects” – Multi value properties only, this is the “Include Any” operator

“Is Null” – Multi value properties only

“Is Not Null” – multi value properties only

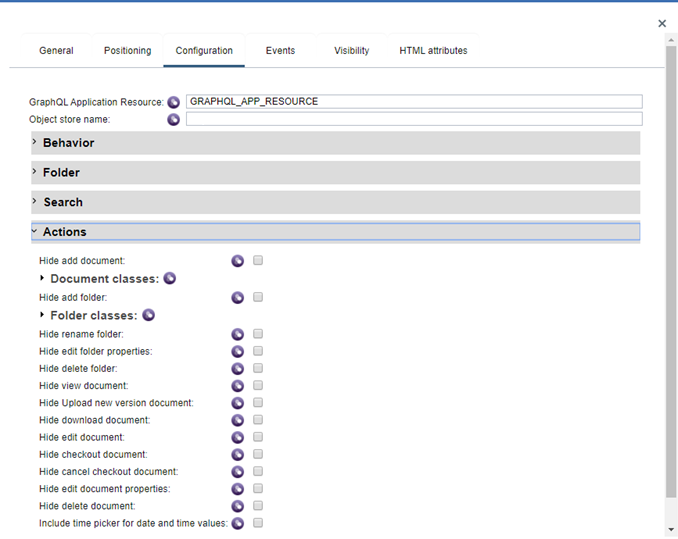


Open the Actions section.

These settings allow the user to hide any of the listed actions from the context menu.

Users can also specify a list of classes that will be exposed from the Add Document and Add Folder modal dialogs (instead of retrieving the entire list from the repository).

Nothing needs to be set here.



At this point, you can save your settings (button on application banner bar):

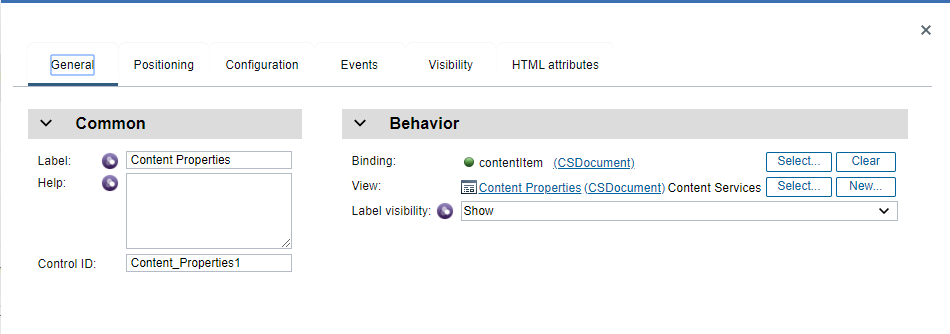


and run your application in test/preview mode (Preview link on far right of banner). You should see the list view rendered with the results from the folder or search you specified.

# Content Properties – Configuration

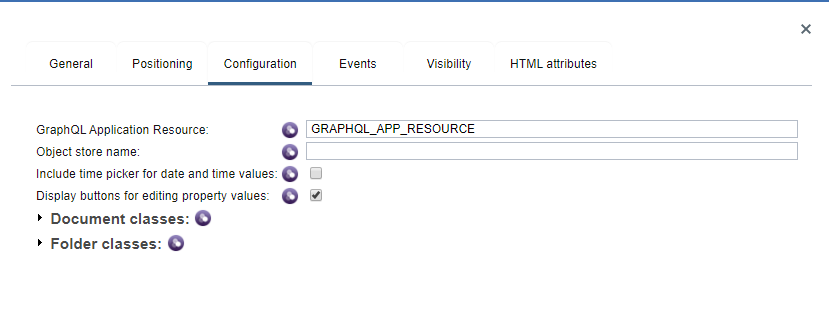
Select the Content Properties component from the palette, place next to the list view (or where ever you’d like it) and open the properties dialog.

On the “General” tab, select the “Binding” property to be the variable called “contentItem”. This variable you created above as part of the when the Content list configuration.



Next, switch to the “Configuration” tab and enter the name of the graphql application resource you created above in the GraphQL Application Resource field.

In the Object store name field, enter the symbolic name of the CE repository.

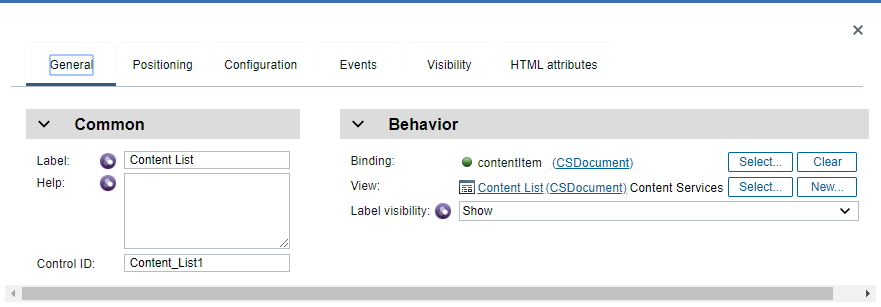


Close the Properties dialog.

Select the Content list and open its properties dialog.

On the “General” tab, select the “Binding” property to be the variable called “contentItem”. This variable you created above as part of the when the Content list configuration.

Both the Content List and the Content Properties components have to be bound to the same application-based variable to enable the event mechanism to work correctly between the two.



Close the Properties dialog and click the save button on the banner bar.

Run your application in test mode (carrot button on far right). You should see the list view rendered and when you select a document from the list view, the property list component will render the document class and class properties.

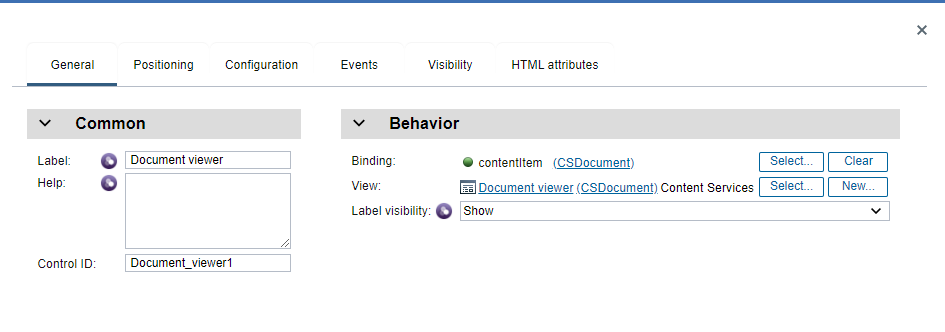
# Document Viewer – Configuration

NOTE: The Document viewer component requires that your application be running inside Navigator with a desktop configured for “Platform” or “Platform and Content” mode. This component uses Navigator’s bookmark.jsp URL running inside an iframe that point’s back to Navigator (it will utilize any viewer map configuration you’ve specified in your desktop). Complete this configuration first, then go on to the following section to setup the viewer to run inside your navigator cloud application.

Select the Document Viewer component from the palette, place below the list view (or where ever you’d like it) and open its properties dialog.

On the “General” tab, select the “Binding” property to be the variable called “contentItem”. This variable you created above as part of the when the Content List configuration.

On the “Configuration” tab set the Object store name to the same settings as your other components.



Close the dialog and Save your settings and run Preview again…

# Document Viewer - Setup

If you plan to use the Document Viewer component in the toolkit, you must update the ESAPIWafPolicy.xml and reinstall it on the Navigator pod.

If you do not complete these steps, when the Document Viewer is invoked, it will not open and you will receive a JS error stating that there is a cross scripting issue.

To update the ESAPIWafPolicy.xml, complete the following steps:

1. In the *<NAVIGATOR CTX ROOT>* and *<APP ENGINE CTX ROOT>* parameters, enter the internet host name or IP address of the Navigator and Application Engine for your local system:

*<NAVIGATOR CTX ROOT>* - Navigator context root without the https: protocal

Example: navigator-bastudio.x.xx.xxx.xxx

*<APP ENGINE CTX ROOT>* - App Engine context root without the https: protocal

Example: playback-bastudio.x.xx.xxx.xxx

Following is the ESAPIWafPolicy.xml file. Note that neither of the strings for the *<NAVIGATOR CTX ROOT>* and *<APP ENGINE CTX ROOT>* parameters are single quoted ‘ when substituted in the file.

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

<policy>

<aliases/>

<settings>

<mode>block</mode>

<error-handling>

<default-redirect-page>/error.jsp</default-redirect-page>

<block-status>500</block-status>

</error-handling>

</settings>

<virtual-patches>

<!-- Uncomment and update to add return host validation when enableOAuthProxy is set to true in web.xml

<virtual-patch id="oauth2-return" path=".\*/jaxrs/oauth2/.\*"

variable="request.parameters.state"

pattern="^(localhost|localhost:http)$"

message="Detected invalid OAuth2 return host.">

</virtual-patch>

-->

</virtual-patches>

<outbound-rules>

<add-header name=*"Strict-Transport-Security"* value=*"max-age=7776000; includeSubdomains"* path=*"/.\*"*/>

<add-header name=*"X-Content-Type-Options"* value=*"nosniff"* path=*"/.\*"*/>

<add-header name=*"X-XSS-Protection"* value=*"1"* path=*"/.\*"*/>

<add-header name=*"X-Frame-Options"* value=*"deny"* path=*".\*/$|^/.+\.jsp$"*>

<path-exception type=*"regex"*>.\*/viewers/[^/]+\.jsp$</path-exception>

</add-header>

<add-header name=*"X-Frame-Options"* value=*"sameorigin"* path=*".\*/viewers/[^/]+\.jsp$"*/>

<add-header name=*"Content-Security-Policy"*

value=*"frame-ancestors <NAVIGATOR CTX ROOT> <APP ENGINE CTX ROOT>; default-src 'self'; connect-src 'self' https:; font-src 'self' data: blob: https:; img-src 'self' data: blob: https:; media-src 'self' data: blob: https:; object-src 'self' https:; script-src 'self' 'unsafe-inline' 'unsafe-eval' https:; worker-src 'self' data: blob: https:; style-src 'self' 'unsafe-inline' https:; frame-src 'self' https:;"*

path=*"/.\*"*/>

</outbound-rules>

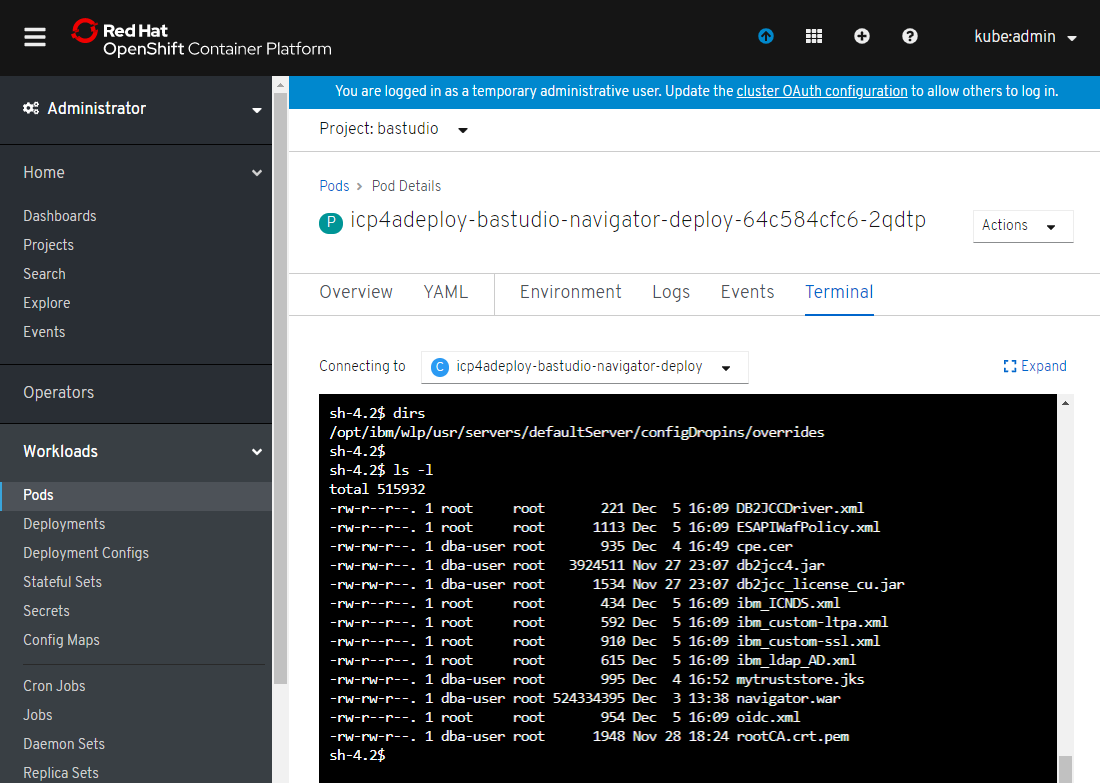
<url-rules>

<restrict-method deny=*"GET"* path=*".\*/jaxrs(/|/.+/)logon$|.\*/jaxrs/pluginRepository$"*/>

</url-rules>

</policy>

1. After you update the file, place it on the cloud system. You can remote to the system or use a tool like “Putty” to log in to the system to add the file. Add the file in the following location: /data/bastudio/icn/overrides
2. Restart the navigator pod.
3. To verify that the file was installed properly, go to the RedHat OpenShift console, and Open Workloads->Pods from left side.
4. Locate the bastudio-navigator pod and open it.
5. Open the “Terminal” tab (in middle on right).
6. “cd” into “configDropins/overrides” folder. The full path is: /opt/ibm/wlp/usr/servers/defaultServer/configDropins/overrides
7. Verify that ESAPIWafPolicy.xml file exists in the folder as shown in the following screen shot:



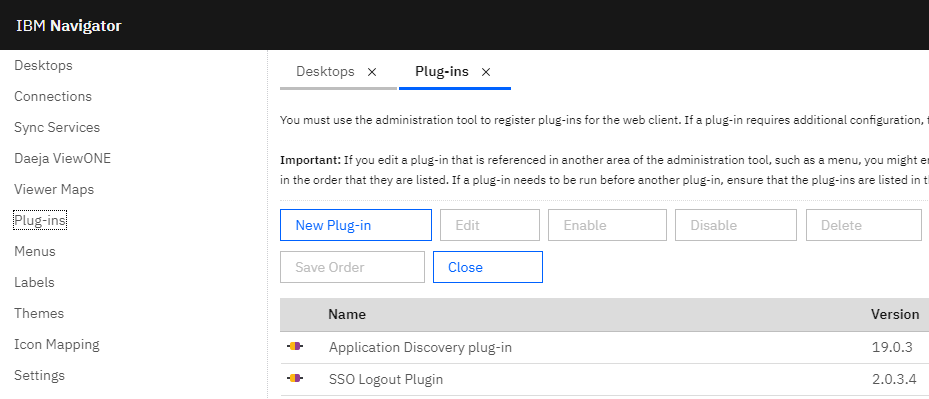
# Navigator – Configuration

Make sure the platform plugin jar is installed and the plugin is enabled into your Navigator environment. The plugin jar file name is:

application-discovery.jar

Run Navigator and login into Administration

Open the Plug-ins feature view and verify the Application Discovery plug-in is running.

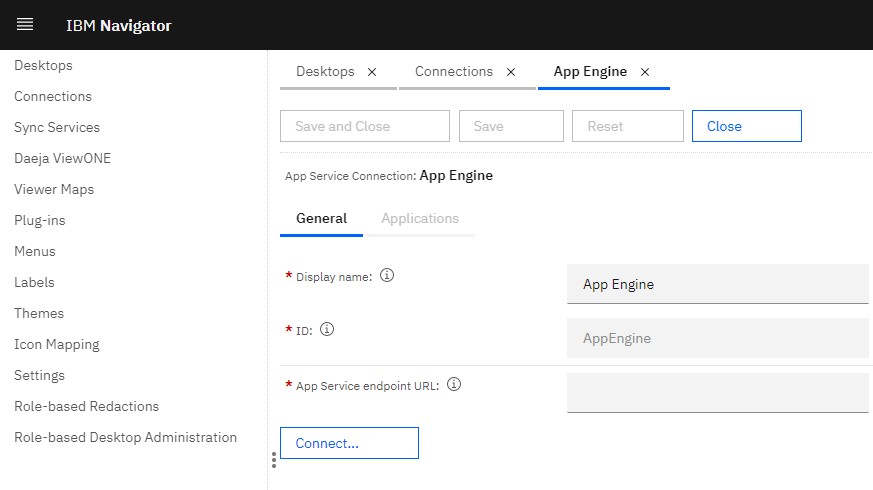


Next under “Connections” feature view, click New Connection button, give it a name:

“App Engine”

Type in App Service endpoint URL to your app server.

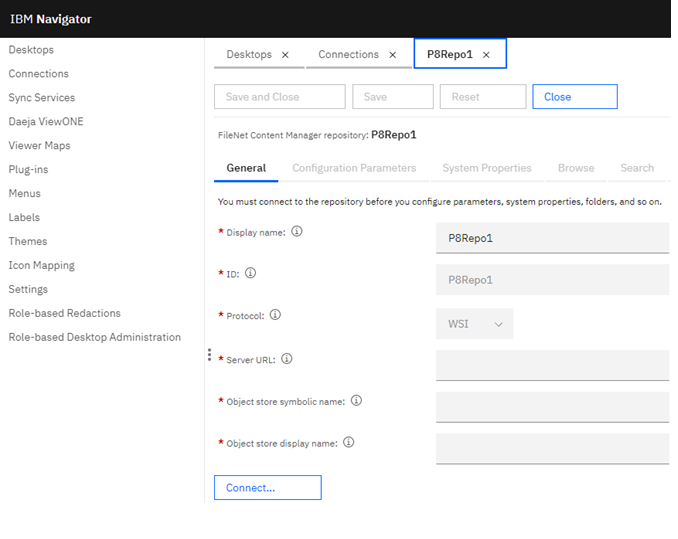
Click Connect…



You’ll also need to have a P8 repository created that will be selected into your desktop.

Enter your repository information, you may need to run the ACCE admin tool to gather the objects store information etc.

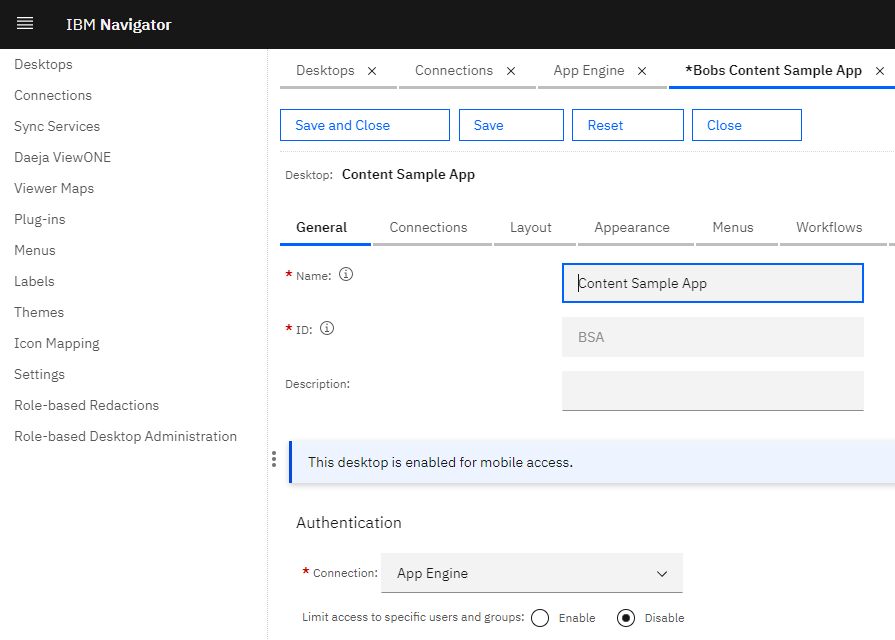
Make sure you can connect by clicking the Connect… button.



Next, open Desktops feature view and create a new desktop.

Point the Authentication “Connection” of your desktop to the connection you just created.

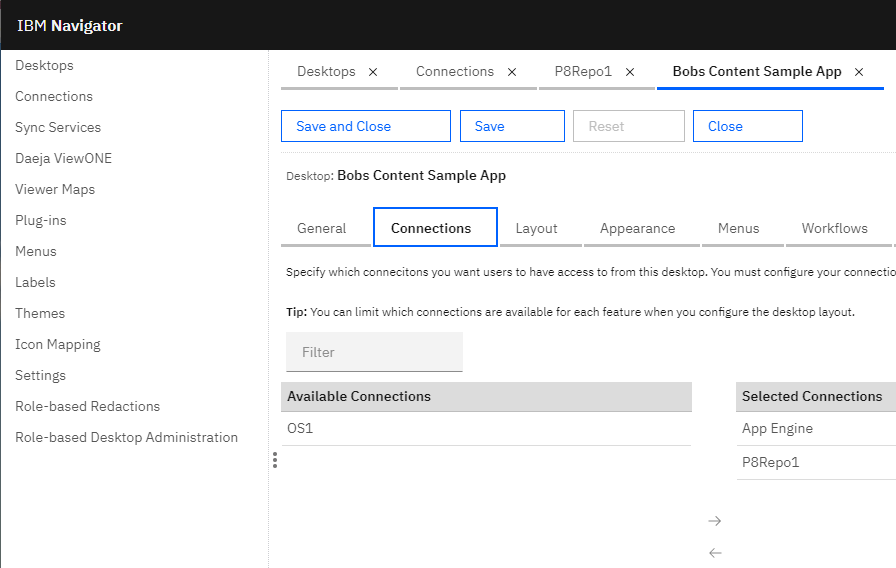
You can also optionally specify a viewer map here too… if you would like to specify a specific viewer.



Next, click on the “Connections” tab.

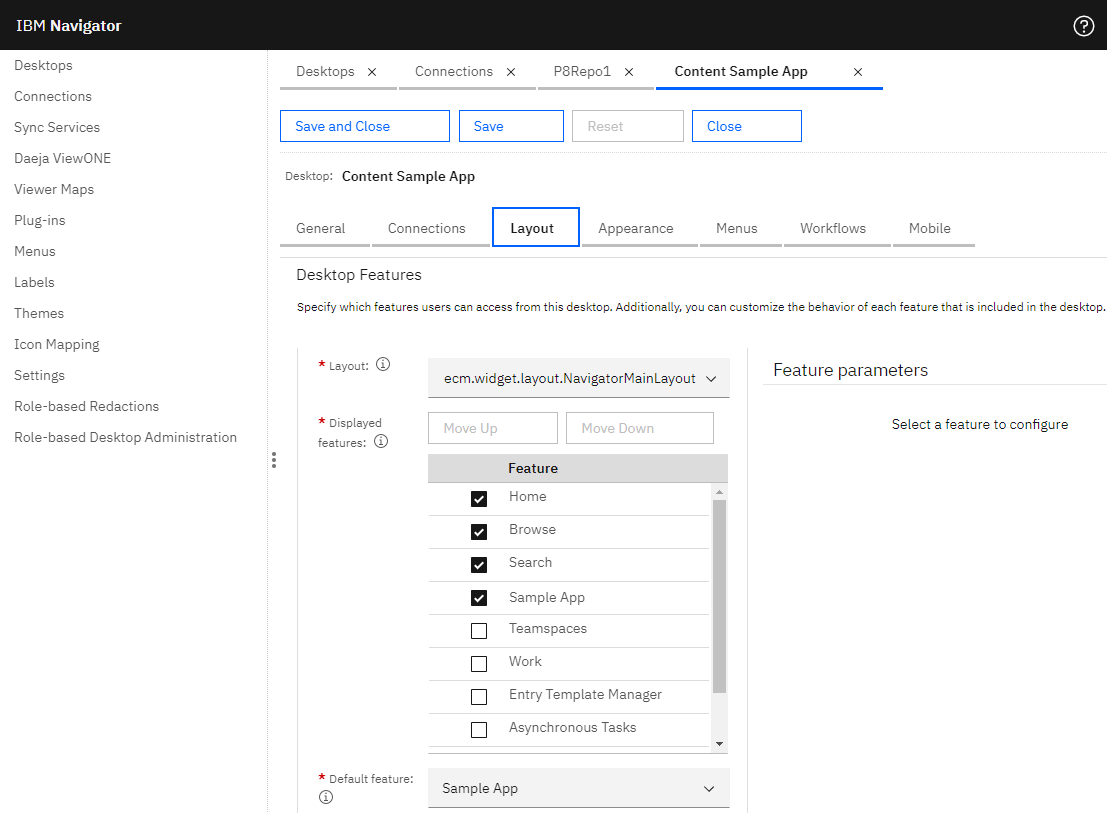
Make sure the App Engine connection is already selected (on right)

Select the Repository where your content will be located, you should have both a connection and repository (a repository is required for the viewer component to work correctly):



Finally, select the “Layout” tab

In the Desktop Features selection area, select your BAStudio application (Sample App in this screen shot) as the feature. You can also have other Navigator features enabled if you are running in “Platform and Content” mode.



Save and login to Navigator using your newly created desktop… Enjoy 😊