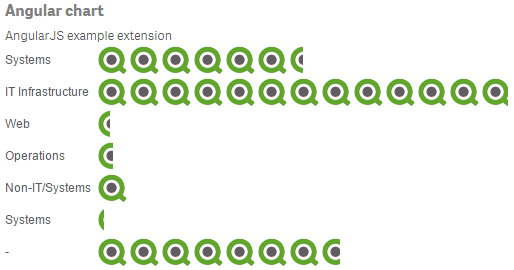
Sample extension (2) Angular chart- when angularJS meet Qlik



In this tutorial, you will learn:

1. Import external resources using requireJS && root directory
2. Loading CSS file and angularJs template
   1. Qlik sense prefix CSS
3. What is” jquery” and “qlik” dependencies
4. Define initialProperties and property panel
5. Using AngularJS in Qliksense extension

What is in the folder

We use Angular chart for our second tutorial. It is easy to identify the chart- it is a bar chart, but the bars are actually Qlik Logo. As the name implies, this extension implements angularJS framework. Finally, we see some programming in action!

Web knowledge is not a must-have for Qlik developers, but it is very useful since Qlik platform was designed and built on top of web programming technology. It is totally fine if you are not a web master now. By the end of all the tutorials, you will have everything a qlik developer needs.

Navigate to C:\Users\[username]\Documents\Qlik\Sense\Extensions\AngularChart, there are 5 files in this folder:

|  |  |
| --- | --- |
|  | Css file defining style |
|  | Main js file |
|  | Jason metadata file |
|  | Template used for angularJS |
|  | Workbench load file |

In this tutorial, we focus only on these three files: js, css and html, mainly on js file since it is the core. More information about qext and wbl can be found in [the hello world tutorial].

The css file and template.html file is new to us, but web developers should be familiar with them. Like in web programming, CSS files define style for extension –colour, layout, font- all the UI related things. Template.html contains Angular-specific elements and attributes. In AngularJs, template is the front-end design module, server combines template with model and controller to render the view.

More information on AngularJS template and CSS:

<https://docs.angularjs.org/guide/templates>

<http://www.w3schools.com/css/css_intro.asp>

What is in the code:

1. Import external files (dependencies) (line 1-2)

We mentioned that Qlik uses requireJS to load modules in [hello world tutorial]. The first two lines of com-qliktech-angularchart.js demonstrates how to use requireJS to load external files.



Explaining the code:

In this example we load 4 external resources into our extension. JQuery, template.html, Css file and qlik library. By defining the dependencies in array (first parameter), you ensure these resources are loaded and that the return value will be passed as parameters in the function.



So in this example, jquery library is loaded as “$”.

"text!extensions/com-qliktech-angularchart/template.html" loaded as “template”.

"text!./angularchart.css" is loaded as “cssContent”;

And “qlik”- root API library is loaded as “qlik”.

Either absolute path or relative path file loading is supported, the root directory is “C:\Users\ [user name]\Documents\Qlik\Sense” by default.

1. Loading Css files and AngularJS template





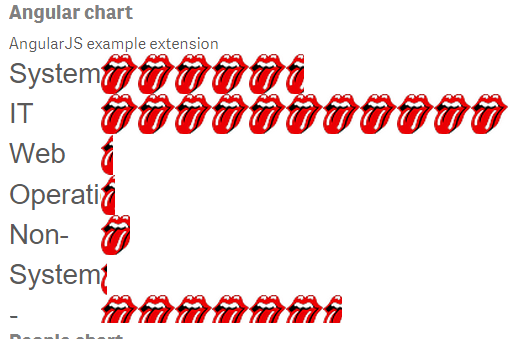
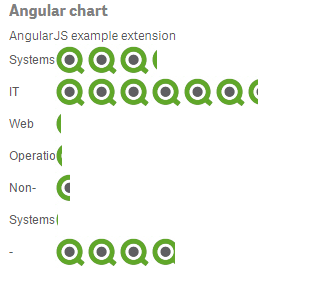
Using “text!” prefix (line 1), we load the Css file the template.html file as plain text.

Angularchart.css is loaded as variable cssContent and append to the header at line 5. 

* $("<object>") is a jquery method to find an existing object in webpage or create a new object if it is not found. In our case "<style>" object is return.
* The content of the cssContent variable is then assigned to the inner content of style object, with format recognized as html.
* The style object, now including the CSS content, is added to the <head> section of the current document.

By doing this, we inject the styles we defined in CSS into the global file. It is always in the header so browsers read them first and apply it during rendering.

If you are new to Css, play around with the attributes to see the effect. Here I changed label font size (line 9) and the background image (line 19). In our example the image was converted to raw format (data Uri) for quick loading.

 (Modified)(Original)

\*Note most browser cache the Css and template content so modification won’t take effect immediately. The easiest workaround is using Qlik sense desktop and reload your app.

Other than manually loading CSS content as text and append into header, it is also possible to load Css file by adding a link to the document style sheet or using requireJS “css!” prefix.

More information about loading Css file can be found here:

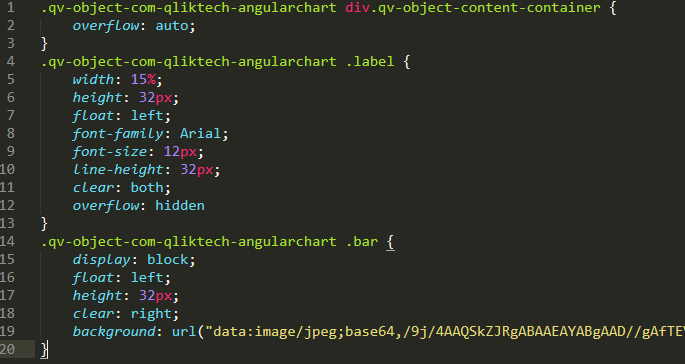
http://help.qlik.com/en-US/sense-developer/2.2/Subsystems/Extensions/Content/extensions-load-resources.htm

template.html (file name) is loaded as “template “(variable name) at line 2 and assigned to template (property for angularJS) at line 7. Kind of confusing, take your time to read it several times. AngularJS framework is one of the third party technology Qlik implement at the backend. Developers can use it to render views instead of using paint method.



* 1. Qlik sense prefix CSS

Now let’s take a step back and look at angularchart.css file. There are three elements in the Css file, all of them start with “. qv-object-com-qliktech-angularchart”, why so?



Css definition is appended to global stylesheet. Therefore, Qlik create CSS class for each extension at the to identify elements in different extension. The name is always in the format of qv-object- [extension name]. Customization/creation of new style must be implemented to this “hidden” class.

Take note that extension name is not the display name that we set in qext file, it is the name of folder.

\*Tips for new developer: the “.” selector at the beginning of element means “apply to all element in this class”. For example, the first part defines the property for the content-container that is in ‘div’ class, but limit to this extension only.

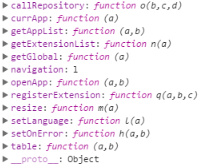
http://www.w3schools.com/cssref/css\_selectors.asp

https://help.qlik.com/en-US/sense-developer/3.0/Subsystems/Extensions/Content/Howtos/working-with-styling.htm

1. “jquery” and “qlik” dependencies

In the dependency array (line 1), there are two special resources  and  which are not referred by path. It is because jQuery and qlik library are pre-configured as internal dependencies of Qlik sense, so there is no need to explicitly include these two files.

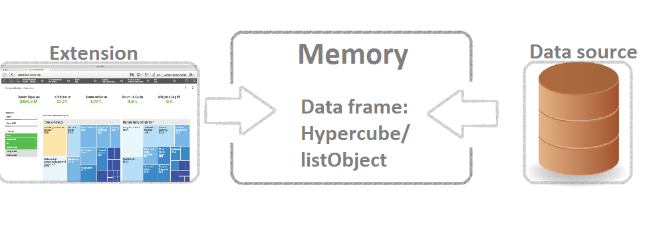
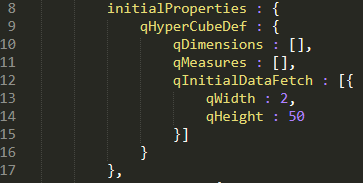
“qlik” library is also known as the Root API, it is used for extensions, mashups and for including Qlik Sense objects in external web pages. As the name suggest, it is the primary entry point for app level APIs and functions.



\*Extension level operations (Data manipulation, selection, actual data etc.) is implemented in backendAPI, which is accessed through “this.backendApi”.

More information on qlik APIs for extension can be found here [Tutorial appendix. Qlik APIs, what are them and how to access them in extensions].

1. Define initialProperties and property panel



initialProperties(line 8-17) defines the initial state of data object when first created. It can be a hypercube or a list object. In simple words, it sets the structure and volume of data in memory. Most Qlik sense built-in visualization are based on a hypercube definition which contains both dimensions and measures.

In this example, we define that no dimension and measure should be set when extension is created and the size of data to be fetched is 2\*50(column\*row).

qDimension is an array of NxDimension.

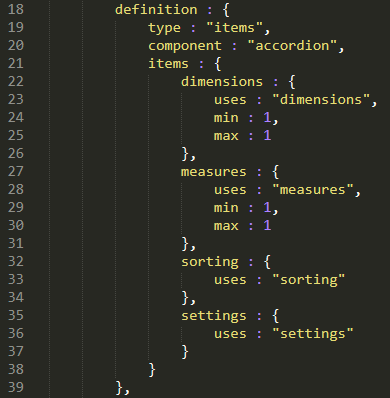
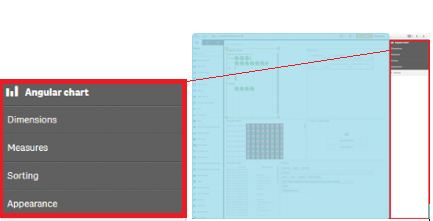
qMeasures is an array of NxMeasure.

And qInitialDataFetch is an array of NxPage.

More details of NxDimension, NxMeasure and NxPage can be found here:

<https://help.qlik.com/en-US/sense-developer/3.0/Subsystems/Extensions/Content/Overview/qHyperCubeDef.htm>.

For explanation on dimension, measure, hypercube and listObject please refer to [Apendix2 dimension measure listObj and hyperCube].

In definition object (line 18-39) we set the property panel of the extension. Property panel is located at the right side of Qlik sense edit page. In our example, we defined 4 items here. By using keyword “uses”, we reuse 4 predefined internal properties of Qlik.: Dimensions, measures, sorting and setting.  These are the out of box properties which is pre-configured. You may extend these pre-defined properties by adding extra attributes, or even define your own property!

Reusing property:

<https://help.qlik.com/en-US/sense-developer/3.0/Subsystems/Extensions/Content/extensions-reusing-properties.htm>

Adding customized property

<https://help.qlik.com/en-US/sense-developer/3.0/Subsystems/Extensions/Content/extensions-make-dynamic.htm>

https://help.qlik.com/en-US/sense-developer/3.0/Subsystems/Extensions/Content/extensions-add-custom-properties.htm

In our example, we limit the number of dimension and measure to one each by setting the min and max value.



And in case you are wondering, “Setting” property was renamed to “Appearance” in the panel.

1. Using AngularJS

AngularJs is a web programming framework, it simplifies data binding between frontend and backend; enables dynamic web programming.

AngularJS

<https://angularjs.org/>

Here is an awesome tutorial on building angularJS Qlik extension from scratch.

<http://help.qlik.com/en-US/sense-developer/3.0/Subsystems/Extensions/Content/extensions-angular-introduction.htm>

Qlik sense is built on top of angularJs; developers may use it to render extensions instead of paint method in extension API. All we need to do is define the template (line 7) for front view, define controller (line 48) for event handling and link the template (frontend view) to model which is in Qlik (line 44 to 46).

Explaining the code (for non-web developer):

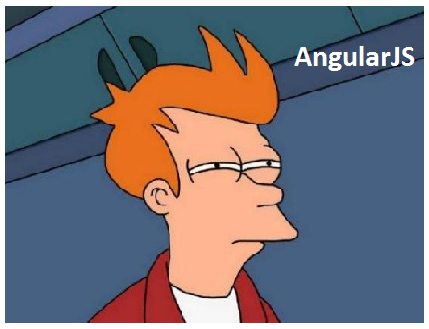
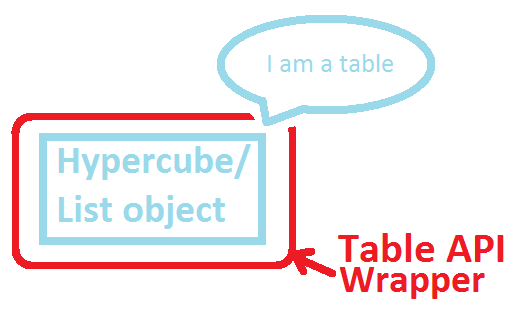


At line 7, we define the angularJs template(https://docs.angularjs.org/guide/templates) by assigning imported template.html (after colon) to angularJs template (befor colon).



In paint method (line 45-46), we create a table object in $scope to qlik data. $scope is the common ground holds variales and functions accessible to both view and controller; think of it as a bridge or something between frontend and backend.

Here we check if the table object exist in $scope; if it is not found, create a table object and pass qlik.table(this) to it. Here we use Qlik table api for data retrieving and handling. Data is wrapped as a table to allow angularJS use iteration functions, like ng-repeat, ng-class etc.



More information on table API please refer to [Tutorial appendix 1. APIs] and our website below:

<https://help.qlik.com/en-US/sense-developer/3.0/Subsystems/APIs/Content/TableAPI/qlik-table-interface.htm>.

At line 52 we define an empty controller to complete angularJS definition, and we will conclude this tutorial here. Following are some of the things I encourage you to try, it will be helpful in understanding Qlik structure.

1. Modify the elements in CSS file and see how UI changes.
2. Is there a way to define $scope.table outside of paint method.
3. Can you remove properties like  in definition? Does it affect anything?
4. Is it possible to define style of other extensions?

Further reading

Angular JS with Qliksense

<http://help.qlik.com/en-US/sense-developer/3.0/Subsystems/Extensions/Content/extensions-angular-introduction.htm>

Styling your visualization

<https://help.qlik.com/en-US/sense-developer/2.2/Subsystems/Extensions/Content/Howtos/working-with-styling.htm>

Using CSS frameworks out of the box

<https://help.qlik.com/en-US/sense-developer/3.0/Subsystems/Mashups/Content/Howtos/mashups-use-css-out-of-the-box.htm>

Loading resources

<http://help.qlik.com/en-US/sense-developer/2.2/Subsystems/Extensions/Content/extensions-load-resources.htm>