Single Page Applications in Web Experience Factory

For customers who want to work with JavaScript in their IBM Web Experience Factory (WEF) portlets, we posted a number of samples on the WEF wiki in January, along with a Script Application builder that automates one particular pattern for script-based portlets. This article will talk about some changes that were made to this builder and a new wizard that was created to make creating and using single page applications easier. For more information on these applications see http://en.wikipedia.org/wiki/Single-page_application. We've now posted an updated version of the Script Application builder as open source, using the OpenNTF repository. It's available here:http://openntf.org/main.nsf/project.xsp?r=project/Script%20Application%20Builder%20for%20IBM%20Web%20Experience%20Factory.

So what have we done to make this easier. Well we made some minor changes to the Script Application builder to make it more obvious that this builders purpose is not just for adding scripting libraries to your pages but can be used to reference existing HTML files and create pages that can be added to your portal. The builder always could do this but it wasn't obvious from the user interface. So why would you do this? Well there are many ways to build single page applications and most these you will end up with a folder that contains css, js, image and html files that are commonly in an app folder that implement the application. But how would you take this and turn it into a portlet? Well you could copy the folder into your project but how do you get the portlet page and portlet. Well to do this you could create an empty model and add two builders to it a Portlet Adapter, which will make the model available on the portal server. And a Script Application builder to reference the index.html file of the application. Doing this will allow the application to run as a portlet. But there is more to it than that sometimes, There can be issues depending on how the application was originally constructed. Sometimes there might be issues with references to components, that were not relative to the new structure of the folder layout(they may need to be relative to the webcontent directory. Or the App might also include some HTML that is incompatible with the processing WEF does on the server. An example of this are applications that use ERB based templating. ERB templating encoding uses an encoding that causes issues with the standard jsp processing that our applications use. There are ways to get around these issues which will be discussed latter in the article. But generally you can get an existing app copy the app folder into your project and create a portlet.

But what is WEF great at? Taking a pattern and automating the repetition of that pattern. So we created a wizard. Which has been added to our OpenNTF offering. This wizard allows to create a new portlet based on an existing application or a blank application for your creation. The wizard asks if you want the model to be a portlet, if you want to include WEF services that are available from an existing service provider and if you want to create a new application or reference an existing one as is. If you choose a new one it asks which libraries you plan to use what application template you want to use and it creates a new model that is available for your use. If you say you want to reference an existing HTML file, you are then asked if it is one that is currently in the project or you have a zip file that contains the application. If it is an existing file then you select the file and the model is created for you. If you choose the zip file you are asked to choose the zip file. The chooser is limited to files that are already in the project by default, but there is a button that allows you to add a file to the project. You can use this to reference zip files that you have downloaded or received from other places. There is also an option to delete the referenced zip file after the wizard has extracted the files into your project. There is also a convert any templating check box that is included to help you convert your applications that need this as they are imported. The most common issue we have run into so far is the default ERB templating that underscore.js supports. So the defaults for the conversion fields are set to convert the settings that it uses. This is done by finding and replacing the underscore.js file with one that has mustache style templating and also finding the templating in the HTML files and also updating those to the mustache style formatting. This format doesn't seem to cause any issue with our jsp processing. There are also some other issues with WEF's processing of HTML as we import it to create pages especially related to relative link references that are sometimes found in these types of applications. We have already fixed these and they will be available in a future release of WEF.

So if you already have some single page applications or are interested in creating some this is one of the easiest ways you currently have to build portlets and consume data services for these types of applications. Please add any comments or issues that you become aware of we are looking to make these better and more useful. For any of these OpenNTF posts, you can download the package from the OpenNTF project page and you can click the "Source Control" link to access the source code on Github. And we'd love for you to get involved. You can post feature requests, report defects, and take part in discussions. If you are interested in posting your own samples or extensions as open source (possible based on something we've posted), for WEF or for any part of Portal, this OpenNTF page has information about how you can contribute: http://openntf.org/main.nsf/page.xsp?name=Get_Involved