Overview

When adding a page to Share, you will probably want your page to match Share's look and feel. In order to do this we will need to discuss Surf Regions, Surf Components, and Alfresco Share's base template that is used to standardize the definition of a Share Surf template. If you want to learn how to add a page to Share, see my previous post Alfresco Share Customization – How To Add a New Page.

Surf Regions

Conceptually a web site has pages. Similar layouts in the pages can be identified and standardized into Surf templates. Each region of the page can be defined as a Surf region. Regions are defined in templates using a Freemarker macro. They can have global, template, or page scope. Here are some examples of global, template and page scoped region tags that you might see in a Surf template:

```
<@region id="header" scope="global">
<@region id="navigation" scope="template">
<@region id="content" scope="page">
```

You can see the idea from the ids of these regions that a global region is something that you would see on every page in the site like headers and footers. Template scoped regions are regions that are included in more than one page consistently like navigation. Page scope regions are unique to a single page.

Components

Components are pieces of content that are defined by REST URL. Components are tied to Surf regions using component bindings, which are found in **/web-extension/site-date/components**. Components can be reused in any number of pages or templates. The URL that a component is tied to can be any RESTful API including webscripts. Here are several examples of component definitions for global, template, and page scoped components:

```
<?xml version='1.0' encoding='UTF-8'?>
<component>
  <scope>global</scope>
  <region-id>header</region-id>
  <source-id>global</source-id>
  <url>/components/header/header</url>
</component>
```

This would be defined in a file located in **/Demo-Share/config/web-extension/site-data/components** and **called global.header.xml**. The scope element defines the scope of the page, **`global`**, **`template`** or **`page`**. The **`region-id`** element corresponds to the region id, as defined in your template region macro (examples given above in the Regions section). The **`source-id`** element corresponds to **`global`** for global scoped components, the template instance id for template scoped components, or page id for page scoped components. Here is an example of a template scoped component:

```
<?xml version='1.0' encoding='UTF-8'?>
<component>
  <scope>template</scope>
  <region-id>navigation</region-id>
  <source-id>home-three-column</source-id>
  <url>/demo/components/navigation/navigation</url>
</component>
```

Here is an example of a page scoped component:

```
<?xml version='1.0' encoding='UTF-8'?>
<component>
  <scope>page</scope>
  <region-id>content</region-id>
  <source-id>home</source-id>
  <url>/demo/components/content/content</url>
</component>
```

You may notice that the URL defined in the url element seems repetitive. This is because more complicated websites will have webscripts with similar functionality grouped together into a single directory. An example is a blog. You might have a webscript that shows new blog posts, a webscript that shows popular blog posts, and a webscript that shows all blog posts. It makes sense to group these into a similar looking URL such as these:

```
/demo/components/blog/new
/demo/components/blog/popular
/demo/components/blog/all
```

It is a good practice to get into the habit of grouping your webscripts now so you don't have to go back and refactor and regroup them later.

Alfresco Global Template

Alfresco has created a set of Freemarker files that Share uses to encourage a uniform look and encourage code reuse. **/Demo-Share/WEB-INF/classes/alfresco/template/org/alfresco/include/alfresco-template.ftl** is a Freemarker template that is included with the base Share templates. It defines macros that load a uniform base of included JavaScript and CSS files common to all Share Surf templates and defines Freemarker macros that are used within each individual template to easily layout a page into header, body, and footer elements. We can include other Freemarker templates within our own Freemarker template by using the include directive. The include directive picks up the included Freemarker from the classpath. Here is what it looks like in our Surf "home" template (a Freemarker template):

```
<#include "/org/alfresco/include/alfresco-template.ftl" />
```

The macros that are provided to us in the **alfresco-template.ftl** are **`templateHeader`**,**`templateBody`**, and **`templateFooter`**. The look like this:

```
<@templateHeader></@>
<@templateBody></@>
<@templateFooter></@>
```

Anything that is included within the tag will be placed into the "header", "body", and "footer" sections within the HTML page. This saves us the time of writing boiler plate HTML for each template and it gives our page similar traits as Alfresco Share's native pages.

New Home Template

The first thing we need to is modify our simply "hello world" home template to fit in with Alfresco Share by reusing the **alfresco-template.ftl**. This is what our new **/Demo-Share/config/web-extension/template/com/tribloom/demo/home.ftl** looks like:

```
<#include "/org/alfresco/include/alfresco-template.ftl" />

<@templateHeader>
  <!-- CSS and JavaScript files are linked in here -->
</@>

<@templateBody>
   <div id="alf-hd">
    <@region id="header" scope="global" />
    <@region id="title" scope="template" />
    <@region id="navigation" scope="template" />
  </div>
  <div id="bd">
    <h1>This is just a test page. Hello World!</h1></body>
  </div>
</@>

<@templateFooter>
  <div id="alf-ft">
    <@region id="footer" scope="global" protected=true />
  </div>
</@>
```

Note the inclusion of the **`templateHeader`**, **`templateBody`**, and **`templateFooter`**regions. These define areas in the page for header, body and content using HTML. We have included several **`region`** tags as well. The global scoped **`region`** tags will pick up the Share components. The template scoped regions will need to have components defined. We can reuse the existing webscripts that back the components but we will have to write a bit of XML.

Adding Components

We need to add a component definition for the title and navigation regions that are defined in our new

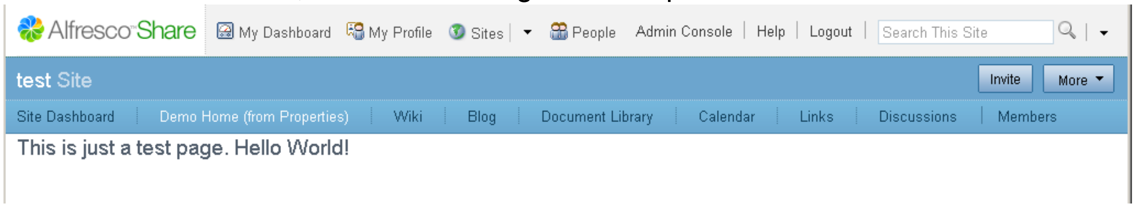
Home template. These regions are defined as template scope in Alfresco's Share pages so we will do the same for our page. The component files should go into **/Demo-Share/config/web-extension/site-data/components/**. We are going to create two new files**template.navigation.home-three-column.xml template.title.home-three-column.xml**. The files follows a naming convention of <global/template/page>.<region-id>.<global/template-instance-id/page-id>. Here is what the **template.navigation.home-three-column.xml** file looks like:

```
<?xml version='1.0' encoding='UTF-8'?>
<component>
  <scope>template</scope>
  <region-id>navigation</region-id>
  <source-id>home-three-column</source-id>
  <url>/components/navigation/collaboration-navigation</url>
</component>
```

The elements in the XML file are analogous the the file naming convention. The **`scope`**element contains **`global`**, **`template`**, or **`page`**. The **`region-id`** element contains the region id that you defined in your template file. The **`source-id`** contains **`global`**, the template id, or the page id depending on the scope of the component. The **`url`** element contains the relative url to the webscript that backs the component. I will leave it up to you to write the other component file (or you can cheat and download the eclipse project).

Wrap Up

When you deploy the new changes and restart your app server and navigate to the home page that we have added, you should get a page that looks very similar to the last project's page, except that this one has Alfresco's header, footer and navigation components!



This is just a test page. Hello World!

Here is a link to the sample Alfresco Share Customization Eclipse project for making your pages look like Share.