BRANDING INTRO

The branding apparatus aims to make it relatively straight-forward to use configurations in order to switch between different variations for displaying the Profiles pages.

In addition to Catalyst and OpenSource configurations, we supply a Foo configuration, representing a fictional (satirical) institution. Foo may be good for experimenting with the features of the branding setup.

> **If you are not sure which branding configuration your are using you can check by running (in a shell) checkConfigurationFolder. For example, if you are currently using the Foo configuration then**
>
> ```
> % checkConfigurationFolder.bash   Foo
> ```
>
> **should indicate three empty diff results.**

For your institution's branding, you can use your own headers and footers, and your own particular values for properties, css definitions and text-snippets, by modifying these files:

1. myBranding.json
2. myBranding.js
3. myBranding.css

For the text-snippets in myBranding.json, you might use an editor with text-wrapping,
as each entry must occur on one (possibly long) line.

EXPLORING THE FOO EXAMPLE

The following examples assume, revolve around, the included 'Foo' branding -- that is the three 'myBranding' files from the Foo folder have been copied up into the Configuration folder.

TEXT-SNIPPETS

Single Snippets: aboutProfiles-whatIsIt

In Foo's myBranding.json we have

> "aboutProfiles-whatIsIt": "Foo \"Profiles\" creates searchable and interactive online profiles of all faculty at Foo University, Foo School of Dental Medicine, and Foo School of Public Health. Foo Profiles is home to several faculty whose expertise covers nearly the entire breadth and depth of biomedicine. But in such a large community, how can you find collaborators to work with on your research? This is where Profiles can help."

In brandingUtil.js, the function loadBrandingConstants() associates the label "aboutProfiles-whatIsIt" with the given text-snippet, using the global variable gBrandingConstants.

> **Note: loadBrandingConstants() looks for your myBranding.json relative to the 'fundamental' value of gBrandingConstants. staticRoot.**
>
> **If the folder that holds your 'static files' is not called StaticFiles, change your myBranding.js near the top:**

```
…
// Set staticRoot to the folder that holds your 'static files', ie
//   the {Coonfiguration, css, html-templates, etc.} folders
gBrandingConstants.staticRoot = "/StaticFiles/";
…
```

As a result, the code can then refer to gBrandingConstants[]. In this case aboutAndHelp.js says

```
$('#whatIsIt').html(gBrandingConstants["aboutProfiles-whatIsIt"]);
```

This code fills in the targeted div in aboutOpenSource.html

```
<div id="whatIsIt"></div>
```

Advanced topic. Grouped Snippets: howToEdit

As an 'advanced' example of text-snippets, you can have grouped blurbs that automatically hide their topic-header if myBranding.json fails to supply values for the blurbs. For example, in help.html, we have

```
...
<div class="topic" sharedAttr="howToEdit">How do I edit my profile?</div>
<div class="blurbForTopic" sharedAttr="howToEdit"
blurb="help-howToEdit1"></div>
<div class="blurbForTopic" sharedAttr="howToEdit"
blurb="help-howToEdit15"></div>
<div class="blurbForTopic" sharedAttr="howToEdit"
blurb="help-howToEdit2"></div>
 ...
```

In Foo's myBranding.json, we have

```
…
"help-howToEdit1": "",
"help-howToEdit2": "",
…
```

In the Catalyst branding configuration, the Json for …Edit1, …Edit15, …Edit2, etc., has non-empty strings. Since the div's in the html use both 'sharedAttr' and 'blurb' attributes, the group or 'systematic' oriented function applySystematicBlurbs(), in brandingUtil.js, will fill in the blurbs UNLESS they are all empty or absent.

Our Foo has no string at all for …Edit15, and empty strings for the other blurbs. So in addition to displaying nothing for those blurbs, the topic paragraph, 'How do I edit my profile?' Is hidden.

PROPERTIES

In common.js we set up favicons on our pages, with the following:

```
let faviconHref = `href="${gBrandingConstants.faviconUrl}"`;
```

```
if (faviconHref) {
  let head = $('head');
  head.append(`<link rel="icon" type="image/x-icon" ${faviconHref}>`);
  head.append(`<link rel="shortcut icon" type="image/x-icon"
${faviconHref}>`);
}
```

This code utilizes the faviconUrl supplied in Foo's myBranding.json

```
"faviconUrl": "https://fooCollege.edu/favicon.ico"
```

Sadly, that particular favicon does not exist, but yours should.

CUSTOM CSS

In Foo's myBranding.css, you can see the setup for the background-color etc. used in Foo's headers and footers.

CUSTOM IMPLEMENTATION FOR REQUIRED FUNCTIONS

In myBranding.js, you may supply your own implementation of

```
setupHeadIncludesAndTabTitle()
emitBrandingHeader()
emitBrandingFooter()
```

For instance, Foo's emitBrandingFooter() sets up 'larger' and 'smaller' footer div's, providing potential for 'responsive' differences between the appearance on larger and smaller screen-widths. You could simplify it to use just one div.

Indeed, the current Foo function treats both large and small footers the same. You could use different text for large/small either by baking it directly into emitBrandingFooter(), or indirectly, by supplying large and small blurbs in the Json, and then accessing via gBrandingConstants.

OTHER RESOURCES

You might also supply other resources, e.g., images that your own *.js or *.html can reference. These should get placed in appropriate folders. E.g., there is a folder …./img/branding, …./js, etc.