Exercises: Managed Beans III

Make a new Dynamic Web Project based on jsf-blank or on the previous set of exercises. Problem 1 (session-scoped beans) is the really important one, but is also a bit more difficult than most previous exercises.

1. Make a form that collects a preferred foreground and background color. Display all of the pages in the a-b-c-victory-defeat app in those colors. Ignore empty strings, and don't let the foreground and background colors be the same. Use a separate session-scoped bean for the color preferences, rather than adding colors to the existing navigation beans (why?). Note that h:body does not support the "bgcolor" and "text" attributes, but it does support the "style" attribute. For example:

```
<h:body style="color: red; background-color: yellow;">
```

You could also build a "style" tag inside h:head. For example:

```
<style type="text/css">
  body { color: red;
         background-color: yellow; }
</style>
```

On your exercises, build up incrementally. First, put in a "style" tag with explicit values (e.g., foreground red and background yellow as above). Run the app and make sure the page comes out red on yellow. Then, make a ColorPrefs class with a getForeground method that returns blue and a getBackground method that returns green. Declare that class with @ManagedBean and change the "style" tag to use the bean, as below:

```
<style type="text/css">
  body { color: #{colorPrefs.foreground};
         background-color: #{colorPrefs.background};
  }
</style>
```

Then, rerun the app and make sure the page comes out blue on green. Finally, after all of that is working, make the bean session scoped, give the user a form to change the properties, verify that the user-entered values affect the page color, and check that the colors persist from page to page.

2. Make a form that collects a search query and randomly shows either the Google results (https://www.google.com/#q=blah) or the Bing results (http://www.bing.com/search?q=blah). Note that this is *far* simpler than the search engine example from the lecture: all you need is a simple SearchController class with a single searchString property and an action controller method. Set the search string to URLEncoder.encode(searchString, "utf-8") in case the search string has spaces or special characters, but there is no need to have a special case for empty strings.