7. Themes

PrimeFaces is integrated with powerful ThemeRoller CSS Framework. Currently there are 30+ pre-designed themes that you can preview and download from PrimeFaces theme gallery.

http://www.primefaces.org/themes.html

7.1 Applying a Theme

Applying a theme to your PrimeFaces project is very easy. Each theme is packaged as a jar file, download the theme you want to use, add it to the classpath of your application and then define primefaces.THEME context parameter at your deployment descriptor (web.xml) with the theme name as the value.

Download

Each theme is available for manual download at PrimeFaces Theme Gallery. If you are a maven user, define theme artifact as;

```
<dependency>
      <groupId>org.primefaces.themes</groupId>
      <artifactId>cupertino</artifactId>
      <version>1.0.8</version>
</dependency>
```

artifactId is the name of the theme as defined at Theme Gallery page.

Configure

Once you've downloaded the theme, configure PrimeFaces to use it.

```
<context-param>
      <param-name>primefaces.THEME</param-name>
      <param-value>aristo</param-value>
</context-param>
```

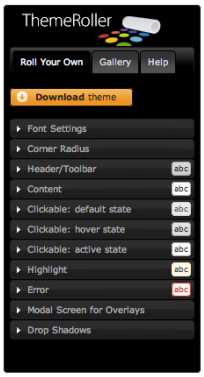
That's it, you don't need to manually add any css to your pages or anything else, PrimeFaces will handle everything for you.

In case you'd like to make the theme dynamic, define an EL expression as the param value.

```
<context-param>
      <param-name>primefaces.THEME</param-name>
      <param-value>#{loggedInUser.preferences.theme}</param-value>
</context-param>
```

7.2 Creating a New Theme

If you'd like to create your own theme instead of using the pre-defined ones, that is easy as well because ThemeRoller provides a powerful and easy to use online visual tool.



Applying your own custom theme is same as applying a pre-built theme however you need to migrate the downloaded theme files from ThemeRoller to PrimeFaces Theme Infrastructure. PrimeFaces Theme convention is the integrated way of applying your custom themes to your project, this approach requires you to create a jar file and add it to the classpath of your application. Jar file **must** have the following folder structure. You can have one or more themes in same jar.

```
- jar
        - META-INF
                - resources
                        - primefaces-yourtheme
                                - theme.css
                                - images
```

1) The theme package you've downloaded from ThemeRoller will have a css file and images folder. Make sure you have "deselect all components" option on download page so that your theme only includes skinning styles. Extract the contents of the package and rename *jquery-ui-{version}.custom.css* to *theme.css*.

2) Image references in your theme.css must also be converted to an expression that JSF resource loading can understand, example would be;

url("images/ui-bg_highlight-hard_100_f9f9f9_1x100.png")

should be;

url("#{resource['primefaces-yourtheme:images/ui-bg_highlight-hard_100_f9f9f9_1x100.png']}")

Once the jar of your theme is in classpath, you can use your theme like;

```
<context-param>
      <param-name>primefaces.THEME</param-name>
      <param-value>yourtheme</param-value>
</context-param>
```

7.3 How Themes Work

Powered by ThemeRoller, PrimeFaces separates structural css from skinning css.

Structural CSS

These style classes define the skeleton of the components and include css properties such as margin, padding, display type, dimensions and positioning.

Skinning CSS

Skinning defines the look and feel properties like colors, border colors, background images.

Skinning Selectors

ThemeRoller features a couple of skinning selectors, most important of these are;

Selector	Applies
.ui-widget	All PrimeFaces components
.ui-widget-header	Header section of a component
.ui-widget-content	Content section of a component
.ui-state-default	Default class of a clickable
.ui-state-hover	Hover class of a clickable
.ui-state-active	When a clickable is selected
.ui-state-disabled	Disabled elements.
.ui-state-highlight	Highlighted elements.
.ui-icon	An element to represent an icon.

These classes are not aware of structural css like margins and paddings, mostly they only define colors. This clean separation brings great flexibility in theming because you don't need to know each and every skinning selectors of components to change their style.

For example Panel component's header section has the *.ui-panel-titlebar* structural class, to change the color of a panel header you don't need to about this class as *.ui-widget-header* also that defines the panel colors also applies to the panel header.

7.4 Theming Tips

- Default font size of themes might be bigger than expected, to change the font-size of PrimeFaces components globally, use the .ui-widget style class. An example of smaller fonts;

```
.ui-widget, .ui-widget .ui-widget {
      font-size: 90% !important;
}
```

- When creating your own theme with themeroller tool, select one of the pre-designed themes that is close to the color scheme you want and customize that to save time.

- If you are using Apache Trinidad or JBoss RichFaces, PrimeFaces Theme Gallery includes Trinidad's Casablanca and RichFaces's BlueSky theme. You can use these themes to make PrimeFaces look like Trinidad or RichFaces components during migration.

- To change the style of a particular component instead of all components of same type use namespacing, example below demonstrates how to change header of all panels.

```
.ui-panel-titlebar {
      //css
}
```

or

```
.ui-paneltitlebar.ui-widget-header {
      //css
}
```

To apply css on a particular panel;

```
<p:panel styleClass="custom">
      ...
 </p:panel>
```

```
.custom .ui-panel-titlebar {
      //css
}
```