



# USING THE LIFERAY THEME GENERATOR

Copyright ©2017 Liferay, Inc.  
All Rights Reserved.  
No material may be reproduced electronically or in print,  
distributed, copied, sold, resold, or otherwise exploited  
for any commercial purpose without express written  
consent of Liferay, Inc.

# START YOUR THEMES

- ❖ We're ready to start defining the look and feel of S.P.A.C.E. using a theme.
- ❖ Defining the branding across Liferay will take a few steps:
  - ❖ Creating the theme
  - ❖ Customizing the HTML on the pages
  - ❖ Defining our styles in CSS
  - ❖ Customizing images
  - ❖ Adding any JavaScript we need
  - ❖ Finishing the theme module's configuration
- ❖ This should deal with most of what the design team needs to establish branding.

# LIFERAY THEME GENERATOR

- ❖ We'll first need to create our theme using the installed tools.
- ❖ The two main tools are:
  - ❖ **Liferay Theme Generator:**  
<https://github.com/liferay/generator-liferay-theme>
  - ❖ **Liferay Theme Tasks:** <https://github.com/liferay/liferay-theme-tasks>
- ❖ With these two tools, we'll speed up our development cycle of *create-modify-test*.

## WHAT DO THE TOOLS DO?

- ❖ Liferay Theme Generator is a *Yeoman* generator that is used to create the theme boilerplate.
- ❖ Liferay Theme Tasks is a set of *gulp* tasks used for building and deploying your theme.
  - ❖ These tasks are automatically installed when creating a theme with Liferay Theme Generator.

## YEOMAN GENERATORS

- ❖ There are four generators in the `generator-liferay-theme` package.
  - ❖ `liferay-theme`
  - ❖ `liferay-theme:import`
  - ❖ `liferay-theme:layout`
  - ❖ `liferay-theme:themelet`
- ❖ Here we will focus on `liferay-theme` and `liferay-theme:import`.

## EXERCISE: CREATING THE SPACE THEME

❖ Let's create the starting point for our theme!

1. **Go to** the *liferay* folder you created in the command line or *Terminal*.
  - ❖ **Windows:** *C:\liferay*
  - ❖ **Mac/Linux:** *~/liferay*
2. **Run** the `yo liferay-theme` command.
  - ❖ This command will create the base theme files and install the necessary dependencies for deployment.
3. **Type** *Y* or *N* if prompted to collect data.
4. **Press** *Enter*.

## EXERCISE: FINISHING UP THE SPACE THEME CREATION

1. **Type** to name your theme *Space Program Theme*.
  - This is what administrators see when they look for your theme in Liferay.
2. **Press** *Enter*.
3. **Press** *Enter* to accept the default themeld.
  - This is what Liferay calls your theme internally.
4. **Choose** 7.0 for the theme version.
  - Liferay DXP is built on Liferay 7.0.

## EXERCISE: SELECTING AN APP SERVER

- ❖ Once you are done following the prompts, it will create a new directory containing the theme files and install the *npm* dependencies.
- ❖ Next, let's choose our app server directory:
  1. **Type** in the command line or *Terminal*.
    - ❖ Windows: `C:\liferay\bundles\liferay-dxp-digital-enterprise-[version]\tomcat-[version]`
    - ❖ Mac/Unix: `~/liferay/bundles/liferay-dxp-digital-enterprise-[version]/tomcat-[version]`
    - ❖ Make sure there are no spaces at the end of the path.
  2. **Press** *Enter* to use the default `http://localhost:8080` for the URL.
- ✓ We now have a base theme to customize!



# NOT QUITE FROM SCRATCH

- ❖ We briefly explored that a new theme in Liferay is built from a *base* theme.
- ❖ This theme gives us a starting point with:
  - ❖ Basic structures
  - ❖ Basic styling
  - ❖ Complete UI components
- ❖ From here, we can add and customize to meet our vision.
- ❖ We can also build on top of other themes, not just the base theme.

# LIFERAY THEME IMPORT

- ❖ If developers have an existing theme, which has not been setup to use Liferay Theme Tasks, they can use `yo liferay-theme:import`.
- ❖ This command does almost the same thing as `liferay-theme`, except it pulls in source files from an existing theme created using Plugins SDK.
- ❖ The only prompt will be for the root file path of the theme being imported. This generator will also create a new directory containing the theme files.



# THEME STRUCTURE

- ❖ After running the `liferay-theme` generator, you will find the following file structure.

File	Description
<code>gulpfile.js</code>	Registers tasks from <code>liferay-theme-tasks</code> to your theme
<code>liferay-theme.json</code>	Generated file created by <code>gulp init</code> that stores appserver related configuration
<code>node_modules</code>	Directory where npm dependencies are installed
<code>package.json</code>	Where theme meta-data is defined and npm dependencies are declared
<code>src</code>	Contains source files of theme, similar to <code>_diffs</code> directory in Plugins SDK themes
<code>src/WEB-INF</code>	Meta-data files such as <code>plugin-package.properties</code> and <code>liferay-look-and-feel.xml</code>

## DEPEND ON THESE MODULES

- ❖ You'll notice a file seen in other Node.js projects: `package.json`.
- ❖ This contains useful information:
  - ❖ Project metadata, such as name and description
  - ❖ Version number
  - ❖ List of dependencies
  - ❖ Node plugins
- ❖ If you open it, you'll notice dependencies like this:

```
"liferay-theme-deps-7.0": "*",  
"liferay-theme-tasks": "*"
```
- ❖ Since the `*` indicates using the latest version, this theme will be built on newest dependencies.

## EXERCISE: SIGNING INTO LIFERAY

❖ Let's deploy our base theme!

1. **Go to** *The Space Theme* using `cd space-program-theme` from your *liferay* directory in the command line or *Terminal*.
2. **Run** the `gulp deploy` command.
  - ❖ This will make our base theme available on our Liferay instance.
3. **Open** your browser.
4. **Sign in** to Liferay.

## EXERCISE: DEPLOYING OUR BASE THEME

1. **Go to** *Menu*→*Site Administration*→*Navigation* for the S.P.A.C.E. site.
2. **Click** *Options*→*Configure* next to *Public Pages*.
3. **Click** *Change Current Theme*.
4. **Choose** *Space Program Theme*.
5. **Click** *Save*.
6. **Click** *Go to Site* to see your theme.

# BEHOLD THE SPACE PROGRAM THEME



S.P.A.C.E.

Breadcrumb



## Breadcrumb

Welcome

## Hello World

Welcome to Liferay DXP Digital Enterprise 7.0.10 GA1 (Wilberforce / Build 7010 / June 15, 2016).

Powered By [Liferay](#)

## ON THE SHOULDERS OF STYLED

- ❖ From here, we can start building our complete theme.
- ❖ Recall that the underlying base theme is *Styled*.
  - ❖ *Styled*: Contains the CSS, JavaScript, and everything needed to complete the base theme. *Styled* has a base theme of *Unstyled*.
  - ❖ *Unstyled*: Contains all the HTML structure and CSS class names.
- ❖ As we build, we'll be using gulp to perform some tasks.
- ❖ Let's get familiar with what they do.



## GULP TASKS: BUILD AND DEPLOY

- ❖ All gulp tasks are executed from the root directory of the theme.
- ❖ Here we will focus on the tasks that pertain to building and deploying your theme.

### 1. **Build:** `gulp build`

- ❖ This task compiles all source files into the `build` directory and creates a WAR file in the `dist` directory of your theme.

### 2. **Deploy:** `gulp deploy`

- ❖ The `deploy` task first runs the `build` task and then deploys the generated WAR file to the specified appserver.
- ❖ If your bundle is running, `deploy:gogo` can be used to deploy your theme. This method is faster than the traditional `deploy` task.
- ❖ **Note:** *Only* use one deploy method. If you first deploy with `deploy`, then don't switch to `deploy:gogo`, and vice versa.

## GULP TASKS: WATCH AND INIT

### 3. Watch: `gulp watch`

- The watch task, similar to the `deploy:gogo` task, will only work when your bundle is running, as it communicates with the Gogo Shell.
- It watches for changes to theme source files and will “fast deploy” so that on page refresh your changes will take effect.

### 4. Init: `gulp init`

- The `init` allows you to specify your appserver path for use with the `deploy` task.
- It is also automatically invoked after `yo liferay-theme` and `yo liferay-theme:import`.
- These properties are saved in `liferay-theme.json`, which is created in your theme's root directory.

# GULP TASKS: EXTEND

## 5. **Extend:** `gulp extend`

- The `extend` task allows you to set what base theme you would like to extend your theme from.
- It also lets you add themelets to your theme, which we'll get to later.

```
[10:26:01] Starting 'extend'...  
? What kind of theme asset would you like to extend? (Use arrow keys)  
> Base theme  
Themelet
```

- To change the base theme, select `Base Theme` in the prompt.
- The prompt will then let you select `Styled` or `Unstyled`.
- The other two options allow you to extend from other themes, exposed as npm package, either installed on the system or published to the npm registry.
- `Styled` and `Unstyled` are npm packages that have been published from `liferay-portal` source.

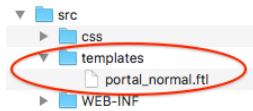
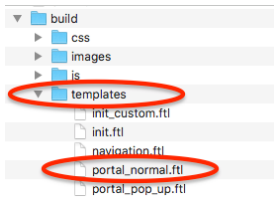
## GULP TASKS: STATUS

### 6. **Status:** `gulp status`

- The status task simply reports what base theme and themelets are implemented.

## ADDING CUSTOMIZATIONS

- ❖ Customizing your theme's CSS, images, and templates is done in the `src/` folder.
- ❖ When you want to modify a file, originally from the base theme, you'll need to copy that file, from the `build` folder, to your `src` folder.
- ❖ Note: the `build` folder gets created after a `gulp build` or `gulp deploy`.



# SASS EVERYWHERE

- ❖ Sass (SCSS) is the language Liferay uses for Stylesheets. Sass, combined with Bourbon's mixins, allows you to use the latest CSS3 features and leverage Sass's syntax advantages, such as nesting and variables.
- ❖ To make use of Sass's syntax and Bourbon, Liferay uses Sass to processes files with the `.scss` extension.

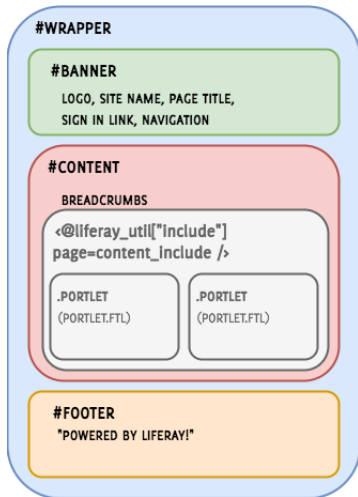
▼	build	Today, 4:55 PM
	.DS_Store	Today, 4:55 PM
▼	css	Today, 4:24 PM
	_application.scss	Today, 4:24 PM
	_aui_custom.scss	Today, 4:24 PM
	_aui_variables.scss	Today, 4:24 PM
	_base.scss	Today, 4:24 PM
	_custom.scss	Today, 4:24 PM
	_extras.scss	Today, 4:24 PM
	_imports.scss	Today, 4:24 PM
	_layout.scss	Today, 4:24 PM
	_liferay_custom.scss	Today, 4:24 PM
	_liferay_variables_custom.scss	Today, 4:24 PM

## KEY FILES FOR THEME CUSTOMIZATION

- ❖ There are some key files to keep in mind when customizing your theme:
  - ❖ `portal_normal.ftl`: The main HTML source file
    - ❖ There are a number of HTML files we can modify, but the `portal_normal` file contains the main structure of the page.
  - ❖ `_custom.scss`: The file used for custom, global styling
    - ❖ Developers can style a number of elements in different files as seen in the `build` folder. These files are combined into one file called `main.css`, which is then linked into the theme's `<head>`, on every page.
  - ❖ `main.js`: Developers should place their custom JavaScript in this file, which is loaded on every page.
  - ❖ `liferay-look-and-feel.xml`: Developers can add different settings, such as *Theme Settings*, *Color Schemes*, and *packaged Layout Templates* in this file.

# CONTROLLING PAGE STRUCTURE

- ❖ Themes have control over every aspect of the page.
- ❖ Every site page in Liferay breaks down to three major customizable sections as we see in the `portal_normal.ftl`:
  - ❖ **The Banner Section:** Includes the top part of a page with its sections
  - ❖ **The Content Section:** Includes breadcrumbs and code to render layouts and applications
  - ❖ **The Footer Section:** Includes the bottom of the page that can be customized





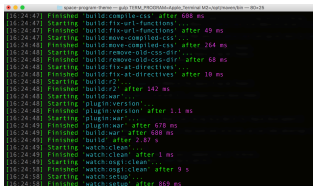
## EXERCISE: MODIFYING OUR THEME

- ❖ We've provided all the necessary file changes for you, allowing us to dive right in.
  - ❖ All of our theme exercise files can be found in *exercises/front-end-developer-exercises/03-theme-development/01-generating-a-theme/exercise-src*.
  - ❖ First, we'll set up our theme's source folders.
1. **Go to** the Space Program Theme `src` folder.
  2. **Create** the following folders in `src`:
    - ❖ `fonts`
    - ❖ `images`
    - ❖ `js`
    - ❖ `layouttpl`
    - ❖ `templates`

## EXERCISE: USING GULP WATCH FOR RAPID DEVELOPMENT

❖ Now that our theme is deployed and selected and we have the basic structure created, we can take advantage of gulp watch.

1. **Go to** your command line or *Terminal*.
2. **Run** the `gulp watch` command.



```
15:24:47 Finished build:compile-css after 688 ms
15:24:47 Starting build:fix-url-functions ...
15:24:47 Finished build:fix-url-functions after 49 ms
15:24:47 Starting build:move-compiled-css ...
15:24:48 Finished build:move-compiled-css after 264 ms
15:24:48 Starting build:remove-old-css-dir ...
15:24:48 Finished build:remove-old-css-dir after 68 ms
15:24:48 Starting build:fix-at-directives ...
15:24:48 Finished build:fix-at-directives after 10 ms
15:24:48 Starting build:r2 ...
15:24:48 Finished build:r2 after 142 ms
15:24:48 Starting build:war ...
15:24:48 Starting plugin:version ...
15:24:48 Finished plugin:version after 1:1 ms
15:24:48 Starting plugin:war ...
15:24:49 Finished plugin:war after 678 ms
15:24:49 Finished build:war after 688 ms
15:24:49 Finished build after 2:87 s
15:24:49 Starting watch:clean ...
15:24:49 Finished watch:clean after 1 ms
15:24:49 Starting watch:uglify:clean ...
15:24:50 Finished watch:uglify:clean after 9 s
15:24:50 Starting watch:setup ...
15:24:50 Finished watch:setup after 869 ms
```

- ❖ Now we can see our changes as we make them.
- ❖ Check Liferay's log output to see when files have been deployed

## COMING FROM LIFERAY 6.2

- ❖ If you've made themes in Liferay 6.2 or earlier, this looks really familiar!
- ❖ Some highlights of Liferay DXP themes versus Liferay 6.2 themes:
- ❖ Themes were created and deployed using the Plugins SDK.
  - ❖ A Plugins SDK is still available, but newer tools with more features have been created to speed up development.
- ❖ Liferay Theme Generator essentially accomplishes the same thing as the Plugins SDK's `themes/create.sh` script.
- ❖ Unlike the Plugins SDK, you can create themes anywhere on your file system.
- ❖ `gulp` is used for running build tasks instead of `Ant`.
  - ❖ Live updating of the theme is available through the `watch` task.
  - ❖ Building from existing themes is easier with the `extend` task.

# THEME STRUCTURE CHANGES

- ❖ There are some specific changes in the structure and details of the theme module:
- ❖ All `.css` file extensions have been changed to `.scss`.
- ❖ Customizations go in a different folder structure:
  - ❖ **Plugins SDK method**
    - ❖ Changes are in the `_diffs` directory.
  - ❖ **Theme-Generator method**
    - ❖ Changes are in the `src` directory.
- ❖ FreeMarker templates are the recommended default:
  - ❖ **Liferay 6.2**
    - ❖ `portal_normal.vm`
  - ❖ **Liferay DXP**
    - ❖ `portal_normal.ftl`

## BUILDING THE THEME'S STRUCTURE

- ❖ With our basic theme module in place, we are free to add our custom brand.
- ❖ Inside the module's source folders, we'll be able to modify:
  - ❖ The HTML structure through templates
  - ❖ The global styling through CSS
  - ❖ Additional styling and branding through images
  - ❖ Custom JavaScript
- ❖ As we build each one, check on the progress in Liferay thanks to gulp Watch.

Notes: