- To make beautiful, responsive, and coherent themes, Liferay 7 uses modern frameworks and languages.
  - SCSS
  - Bourbon
  - Bootstrap 3
  - Lexicon
  - FreeMarker

4.1.4 Gulp

- Gulp is a *Node.js*-based task runner. It is a newer alternative to *Grunt*.
- Tasks are single-responsibility building blocks that can produce a desired output.
- Gulp uses a stream-based architecture.
- Gulp handles file-watching and swapping, compilation, and WAR file build.
- Gulp can be easily installed with npm:
  - `npm install -g gulp`

4.1.5 Yoman

- Yoman is a scaffolding tool. It uses a generator to create new projects.
- Liferay provides an official theme generator. The generator is able to create a new theme and has two sub-tasks for the conversion of the Plugin SDK themes and creating themelets.
- Yoman can be easily installed with npm:
  - `npm install -g yo`
- The Liferay theme generator can also be installed with:
  - `npm install -g generator-liferay-theme`

© 2016 Liferay, Inc. All rights reserved.

4.1.6 SCSS

- Liferay has used SCSS for styling since 6.2.
- Ruby compiler was used in 6.2 for on-the-fly CSS compilation. As this caused problems, it was removed. The SCSS files are now compiled with a Java compiler when the theme is created.
- Liferay 7 uses files with the .scss extension only for development. Those are then compiled to the *main.css* and *aui.css*.
- Large CSS files were split into smaller linked SCSS files.
- Components scss files (base/_logo.scss) → group scss files (base.scss) → main.scss → main css.

4.1.7 JavaScript

- AUI/YUI is currently in use, but development has stopped.
- Liferay 7 contains:
  - jQuery 2.1.4
  - Lodash
  - Lexicon jQuery plugins (contains Bootstrap js)
  - AUI + other third party tools
- jQuery and Lodash are namespaced for Liferay use. Other versions of the frameworks may coexist in Liferay.
  - AUI.$ = window.jQuery
  - AUI._= window._

4.1.8 Bourbon

- Liferay 6.2 used *Compass* for mixins. Liferay 7 uses *Bourbonjs* instead.
- Bourbon is a simple and lightweight mixin library for scss.
- It offers nearly all the functionality as Compass but is more lightweight.

©2016 Liferay, Inc. All rights reserved.

- Bourbon has extensions that provide some of the missing functionality:
  - Neat
  - Bitters

4.1.9 Bootstrap 3

- Liferay 7 uses Bootstrap 3
- Liferay uses only the CSS framework part of Bootstrap
- Many Bootstrap classes names changed from 2 to 3
- Main difference is better support for responsive design
  - The grid systems were extended and specific styling classes for different devices were introduced.
  - Rules for easy change of images

4.1.10 Lexicon

- Liferay Experience Language.
- Lexicon is a bootstrap extension that provides a set of predefined visual components.
- Helps to create a unified look and feel
- This has been well-documented with examples:
  - http://liferay.github.io/lexicon/

4.1.11 FreeMarker

- Freemarker was introduced in Liferay 6.0. Since then, it has been used for theme development.
- Most themes in previous versions of Liferay used Velocity.
- Liferay 7 doesn't recommend using Velocity. Although it can still be used, developers are encouraged to move to FreeMarker.
- The biggest advantage to using FreeMarker is that it works with Java Tag libs.

©2016 Liferay, Inc. All rights reserved.

4.1.12 Themelets

- Themelets are reusable fragments of styling.
- You can upload these fragments into your node js repository and use them to rapidly develop new themes.
- They cannot be used on their own.

4.1.13 Bower

- Bower JS is a packaging manager for the web.
- You can manage all your front-end resources (HTML, CSS, JS) with bower.
- Liferay contains a wide variety of resources out of the box.
- Bower can be used to manage additional resources. Alternatively, you can rely only on Bower-imported resources.

4.1.14 Browser support

- Internet Explorer
  - Dropping support for IE9 and below
  - Partial support for IE10
  - Support for IE11/Edge
- Support for latest versions of Chrome, Firefox, and Safari

Exercise

Create a new Liferay theme using the new development stack. The theme should inherit from the `_styled` default theme.

1. Install the necessary tools.
   a) NodeJS - `https://nodejs.org/en/download/`.
   b) Gulp, Yoman, Liferay Theme - `npm install -g gulp yo generator-liferay-the`
2. **Open** command line and generate new theme by running `yo liferay-theme`.

©2016 Liferay, Inc. All rights reserved.

3. **Use** the interactive generator to create a **Liferay 7** theme that uses **FreeMarker**. You can use **bootcamp-theme** as the theme name.
4. **Provide** the path to the Liferay 7 **deploy** folder and the Liferay home URL.
5. **Go to** the generated folder `bootcamp-theme`.
6. **Deploy** the theme `gulp deploy`.
7. **Go to** the platform and change the theme of the default site.

The new stack offers a fast way to observe changes to the theme.

1. **Open** `_custom.scss` and add a style to color the background to light green.
2. **Redeploy** the theme `gulp deploy`.
3. **Refresh** the page.
4. **Run** `gulp watch`
5. **Change** the background color to light blue and **save** the file.
    - Watch the gulp console. It should log that it's changing files.
6. **Refresh** the page.

4.2 Lexicon

4.2.1 What is Lexicon?

- It is the new HTML and CSS framework used by Liferay.
- Its goal is to offer easy access to beautiful content, yet be customizable and fully extensible.
- Written in Sass
- Uses Bourbon to handle browser prefixing
- jQuery Javascript library
- Built on Bootstrap 3

©2016 Liferay, Inc. All rights reserved

4.2.2 Sass

- Sass File + Sass Compiler = CSS File
- Sass Features:
  - Nesting
  - Variables
  - Loops
  - Functions
  - Mixins
- Learn more at: `http://sass-lang.com/`

4.2.3 Sass: Nesting

- Inputs:

```
#main {
  .content {
    color:blue;
  }

  p {
    background-color: red;
  }
}
```

- Outputs:

```
main .content {
  color: blue;
}

# main p {
  background-color: red;
}
```


©2016 Liferay, Inc. All rights reserved.

4.2.4 Sass: Variables

- Inputs:

```
$baseColor: green;

# main {
  color: $baseColor;
}

.ticket-item {
  color: $baseColor;
}
```

- Outputs:

```
# main {
  color: green;
}

.ticket-item {
  color: green;
}
```

4.2.5 Sass: Loops

- Inputs:

```
@for $i from 1 through 4 {
  .col-#{$i} {
    width: percentage($i / 4);
  }
}
```

- Outputs:

```
.col-1 { width: 25%; }
.col-2 { width: 50%; }
.col-3 { width: 75%; }
.col-4 { width: 100%; }
```

©2016 Liferay, Inc. All rights reserved.

4.2.6 Sass: Functions

- Inputs:

```
@function squared($num) {
  @return $num * $num;
}

.squared {
  width: squared(2) + px;
}
```

- Outputs:

```
.squared { width: 4px; }
```

4.2.7 Sass: Mixins

- Inputs:

```
@mixin border-radius($radius) {
  -webkit-border-radius: $radius;
  -moz-border-radius: $radius;
  border-radius: $radius;
}

.box {
  @include border-radius;
}
```

- Outputs:

```
.box {
  -webkit-border-radius: 3px;
  -moz-border-radius: 3px;
  border-radius: 3px;
}
```


©2016 Liferay, Inc. All rights reserved

4.2.8 Bourbon

- Bourbon is a simple and lightweight mixin library for Sass.
- Without Bourbon:

```
# main {
  -webkit-transition: all 0.5s ease;
  -moz-transition: all 0.5s ease;
  transition: all 0.5s ease;
}
```

- With Bourbon:

```
# main {
  @include transition(all 0.5s ease);
}
```

- Learn more at: `http://bourbon.io/docs/`

4.2.9 jQuery

- jQuery is a fast, small, and feature-rich JavaScript library.
- It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers.
- Providing a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.
- Learn more at: `https://jquery.com`

4.2.10 Bootstrap

- The most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first projects on the web
- Provides unified styles for major browser versions
- Liferay themes use Bootstrap 3
- Learn more at `http://getbootstrap.com/`

©2016 Liferay, Inc. All rights reserved.

4.2.11 Lexicon

Overview

- In the past, it could be confusing to know the best way to make use of HTML, CSS, and JS within Liferay to create beautiful web applications.
- Lexicon was developed to provide a consistent and easy to use API that extends Bootstrap 3 to help developers build UI components in and outside of Liferay.
- Lexicon fills the gap between Bootstrap Framework and Liferay Customer Experience design.

Traditional Hierarchy

- _unstyled
- _styled
- classic

Lexicon Base

- Minimum amount of code to integrate Bootstrap 3 with Liferay
- _styled = Bootstrap + Lexicon Base

How Sass Files are Loaded in Liferay

```
File
theme's aui.scss
|---- _aui_variables.scss
|---- ../alloy-font-awesome/scss/font-awesome
|---- lexicon-base/main
|---- ../../liferay_variables_custom
|---- ../../liferay_variables
|---- ../../aui_custom
|---- ../../liferay_custom

frontend-css-web/main.scss
|---- portal/aui_deprecated.css
```


© 2016 Liferay, Inc. All rights reserved

```
|---- portal
|---- taglib
|---- portal/accessibility
```

- Description
  - Contains Bootstrap, Lexicon/Atlas
  - Overwrites Bootstrap/Lexicon/Atlas variables

```
apps/all main.scss

theme's main.scss
|---- imports
|
|
|---- base
|---- portal
|---- taglib
|---- application
|---- layout
|---- control_menu
|---- navigation
|---- portlet
|---- extras
|---- custom
```

- Description
  - Imports third-party libraries required for theme e.g. Bourbon, Liferay Mixins, Lexicon Base Variables, and Bootstrap Mixins.

Basic Lexicon Elements

- Lexicon has a set of basic elements which are used to create high-level Lexicon components.
- Basic elements available in Lexicon are:
  - Typography
  - Breadcrumbs
  - Figures

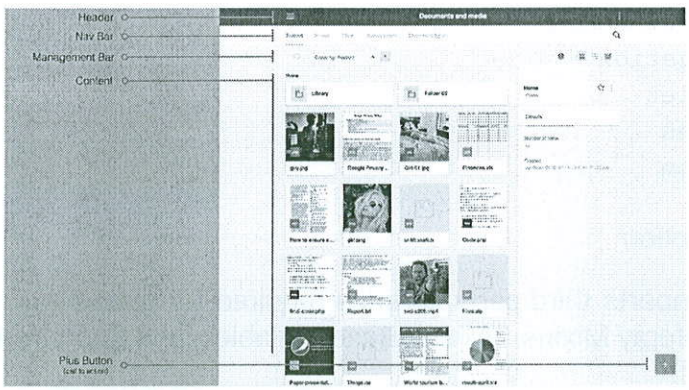
©2016 Liferay, Inc. All rights reserved.

- Icons (Basic)
- Icons (Font Awesome)
- Icons (Glyphicons)
- Images, Thumbnails, and Aspect Ratios
- Nameplates
- Pager
- Sidebar
- Tab Groups
- Tables
- Toolbars

Lexicon Example

(Figure 4.1, page 140)

Figure 4.1:



4.2.12 Lexicon Components

Alerts

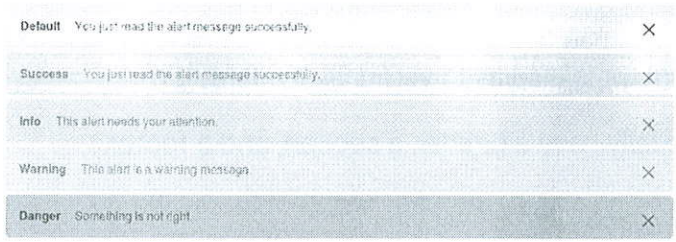
- There are three types of alerts:
- Alerts with Embedded Links
- Dismissible Alerts

©2016 Liferay, Inc. All rights reserved.

- Alerts as notifications

(Figure 4.2, page 141)

Figure 4.2:



Badges, Labels, and Stickers

- Badges help highlight important information such as notifications or new and unread messages.
- Stickers are monospaced badges.
- Labels are Badges with rounded borders.

(Figure 4.3, page 141)

Figure 4.3:

Buttons

Lexicon provides five types of buttons:

- Dropdown Buttons
- Dropup Buttons
- Split Dropdown Buttons
- Primary Action Buttons
- Secondary Action Buttons

(Figure 4.4, page 142)

©2016 Liferay, Inc. All rights reserved.

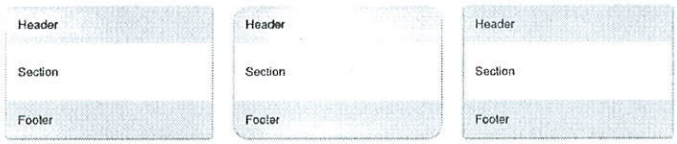
Figure 4.4:

Cards

- Card content is displayed vertically by default.
- They usually have a header, a section, and a footer.

(Figure 4.5, page 142)

Figure 4.5:



Header Toolbar

- Header Toolbar is based on Toolbars.
- Created to quickly prototype headings for portlets, mobile pages, and page sections

(Figure 4.6, page 142)

Figure 4.6:

Icons

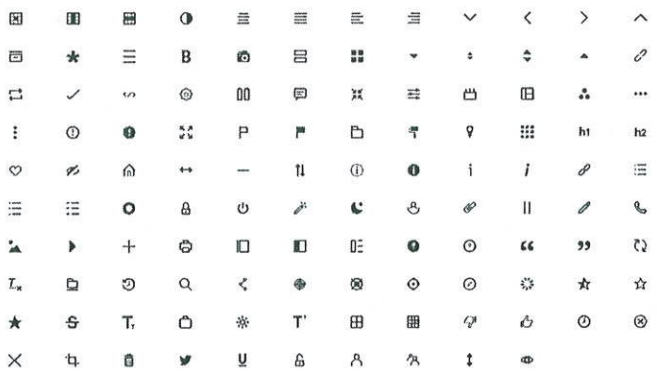
- Lexicon icons do not use the same format as Font Awesome or Bootstrap's Glyphicons.
- Lexicon uses SVG sprites with xlinks to the single icon we want to display.

©2016 Liferay, Inc. All rights reserved.

```
<svg class="lexicon-icon">
    <use xlink:href="path/to/icons.svg#add-column" />
</svg>
```

(Figure 4.7, page 143)

Figure 4.7:



Management Bar

- Extension of Nav Bar
- Combines different Management Bars components to create a custom toolbar that fits your needs

(Figure 4.8, page 143)

Figure 4.8:

Modals

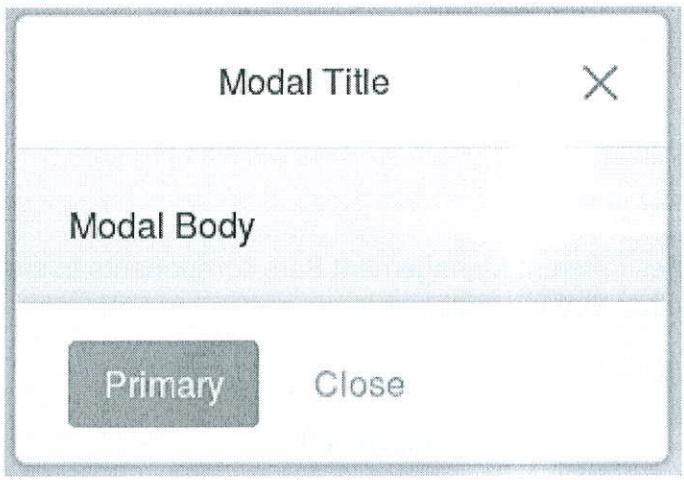
- Lexicon modals have some default sizes:
- Small (300px)
- Default (600px)
- Large (900px)

©2016 Liferay, Inc. All rights reserved.

- Full Screen
- With 5 variants:
- No Header
- No Footer
- Iframe
- Inline Scroller
- Inverse

(Figure 4.9, page 144)

Figure 4.9:



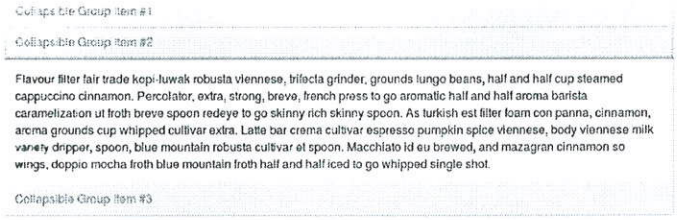
Panels

- Panels help separate your content.
- Panel States: Different colors for different states
- Accordions with Panels and Collapse Plugin: Combine the panel component with collapse to create accordions.

(Figure 4.10, page 145)

©2016 Liferay, Inc. All rights reserved.

Figure 4.10:

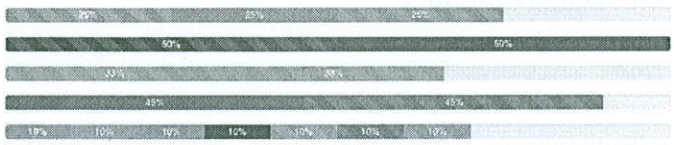


Progress Bars

- Lexicon provides several types of progress bars:
- Multi Step Progress Bar
- Plain Progress Bars
- Progress Bars with Label
- Contextual Progress Bars
- Striped Contextual Progress Bars
- Active Progress Bars
- Stacked Progress Bars

(Figure 4.11, page 145)

Figure 4.11:



Timelines

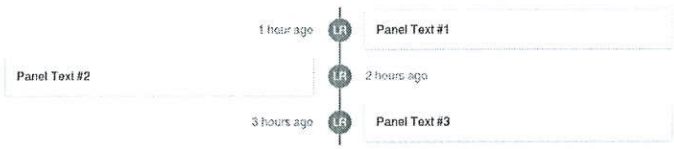
- Lexicon also provides several types of timelines:
- Left
- Right

©2016 Liferay, Inc. All rights reserved

- Center
- Even
- Odd
- Right XS Only
- Spacing

(Figure 4.12, page 146)

Figure 4.12:



Toggles

- Lexicon also provides some toggles:
- Switch
- Switch with Data Attributes
- Switch Text
- Switch with Icons
- Disabled
- Toggle Card
- Toggle Card DM

(Figure 4.13, page 147)

©2016 Liferay, Inc. All rights reserved.

Figure 4.13:

4.2.13 JS Components

- Collapsible search
- jQuery plugin that opens and closes a search bar in mobile.
- Sidenav
- A jQuery plugin that creates a slideout navigation on the right or left for the sidebar css component.
- More information about these JS components:
- http://liferay.github.io/lexicon/content/collapsible-search/
- http://liferay.github.io/lexicon/content/sidenav/

©2016 Liferay, Inc. All rights reserved.

4.2.14 Choosing the right Lexicon Component

- There are many components in Lexicon. Some components overlap, some look the same at first glance, and others are combined to form a whole new component.
- When building your module, be careful not to choose and modify the first component that looks like it may work. Take the time to choose the right component for the job.
- For example, Figures, Aspect Ratio, and Crop Img are components that are similar but serve different purposes. Figures are intended to be used for captioning images, while Aspect Ratio and Crop Img are intended for sizing images. Aspect Ratio maintains a specific ratio relative to its container and Crop Img crops the viewable area of an image instead of changing the image size. **Aspect Ratio and Crop Img can be used in Figures, but Figures shouldn't be used in Aspect Ratio or Crop Img.**

4.2.15 Third Party Bootstrap Theme with Lexicon

- **Create** your new theme.
- **Copy and paste** the sass variables from the third party Bootstrap theme into `_aui_variables.scss` and the styles into `aui_custom.scss`.

4.2.16 References

- Documentation: http://liferay.github.io/lexicon
- Examples:
- `http://liferay.github.io/lexicon/content/blogs-action`
- `http://liferay.github.io/lexicon/content/documents-and-media`
- `http://liferay.github.io/lexicon/content/form-examples`

4.3 Metal JS

©2016 Liferay, Inc. All rights reserved.

4.3.1 What's Metal JS?

- Component framework developed by Liferay
- Built with the use of modern front-end technologies
- Designed with the platform's needs in mind
- Steel UI and Crystal UI are components in development that use the Metal JS framework.

4.3.2 Features of Metal

- Written with the use of EcmaScript6
- Very small overall framework size
- Versatile Build to global objects, AMD, or jQuery Plugins
- HTML Templates
  - Soy Templates
- Progressive enhancement

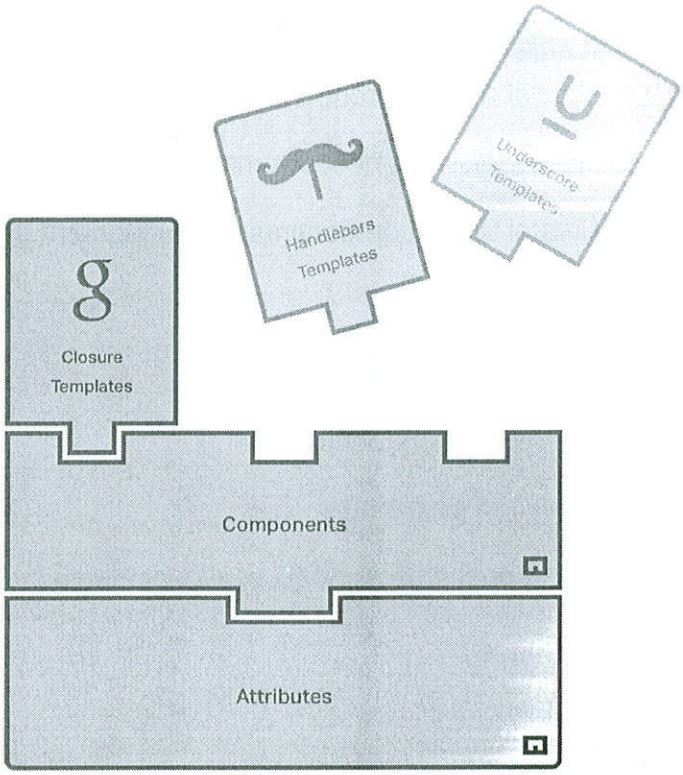
4.3.3 Architecture

- Metal contains two main core classes: Attributes and Components.
- Attributes provide the basic functionality that is needed to create Metal modules.
- Components extend Attributes and provide rendering capabilities for the Metal module.
- The aim is to separate rendering and business logic. The rendering can be done with the help of the component and a template engine.
- Metal currently supports Soy Templates (Closure templates).

(Figure 4.14, page 150)

©2016 Liferay, Inc. All rights reserved.

Figure 4.14:



4.3.4 Soy Templates

- Provided rendering engine for Metal components
- Developed by Google
- Soy Templates are compiled.
- Compilation can be done on server side (Java) or client side (JavaScript).

4.3.5 Metal and Liferay 7

- Liferay does not use Metal components but it contains all the infrastructure that is required for using Metal.
- You can create your own Metal components, compile them into JS, and include them in your projects.

©2016 Liferay, Inc. All rights reserved.

4.3.6 Resources

- The project is well-documented, and there are detailed tutorials that can be found here:
  - `http://metaljs.com/`
- The project is Open-Source and can be found on github.
  - `https://github.com/metal/`
- The Metal GitHub repository contains many Metal components.

4.4 Senna JS

4.4.1 What's Senna JS?

- It's a javascript framework developed by Liferay to improve page load performance.
- It tries to mimic the behavior of Single Page Applications, allowing for dynamic page content changes via AJAX calls.

4.4.2 Features

- Senna JS features are developed with platform behavior in mind.
  - SEO & Bookmarkability
    - URLs should always point to the same content and state of the page.
  - History navigation
    - Makes the browser function usable with History API
  - UI feedback & transition
    - Indicates that something is happening to the User
  - Cacheable screens
    - Content is loaded only once.

©2016 Liferay, Inc. All rights reserved.

4.4.3 How Does it Work?

- Areas of the page are marked to be swapped.
- Senna scans the page for links and takes over when a link is pressed. These processed links are called routes.
- Instead of a traditional request, an AJAX request is made. The response is parsed and the defined surfaces are updated.
- Senna can work automatically, but you can use the framework to define routes and custom responses manually.

4.4.4 Liferay 7 and Senna JS

- Liferay 7 uses Senna JS on all pages to provide a fast browsing experience.
  - The surface is defined on the body level. Portlets are not updated individually.
- Senna JS adds a surface wrapper dynamically. The theme styles need to be prepared for the change.

©2016 Liferay, Inc. All rights reserved.