References

Table of Contents

Session 1: Basics of Bootstrap

Bootstrap for Beginners

Source	http://csharp-video-tutorials.blogspot.in/2016/05/bootstrap-tutorial-for-beginners.html
Date of Retrieval	20/03/2017

Bootstrap is a free, open-source and is the most popular HTML, CSS, and JavaScript framework developed by twitter for creating responsive Web applications. It includes HTML and CSS based design templates for common user interface components such as Buttons, Dropdowns, Typography, Tabs, Forms, Tables, Navigations, Alerts, Modals, Accordion, Carousel etc. along with optional JavaScript extensions. Bootstrap framework is based on open standards - HTML, CSS and JavaScript. This means bootstrap can be used with any server side technology and any platform. You can use it with any Web application built with any server side technology such as ASP.NET, JAVA, PHP etc.

What are the advantages of using bootstrap?

Supports responsive design : One of the greatest advantages of using bootstrap is that it helps us create responsive web applications faster and easier. So the obvious question that comes to our mind is, what is a responsive Web application? A responsive Web application automatically adapts to different screen sizes (i.e desktop computers, laptops, tablets. mobile phones etc). A responsive application provides optimal viewing and interaction experience i.e easy reading and navigation with a minimum of resizing, panning, and scrolling across a wide range of devices. So you don't have to worry about your application not being compatible with multiple devices.

The images at the following links shows how a responsive and non-responsive application looks such as on a mobile device.

http://www.webnersolutions.com/wp-content/uploads/2015/08/responsive-vs-non-responsive-Web-design1.jpg

At the following link is a live responsive application example. Notice as we resize the browser, the content automatically adapts to the screen size. http://bootstrapdocs.com/v3.0.3/docs/examples/jumbotron/

Saves lot of development time : One of the biggest advantages of using Bootstrap is that it saves lot of development time. Instead of writing code from the scratch, bootstrap offers ready made blocks of code that you can use and customize to suit your application requirements. There are also many Websites out there that offer free and paid Bootstrap themes that saves even more development time.

Consistency : Bootstrap was developed by Twitter to encourage consistency across thier internal tools by giving their developers a centralised development code. Since all the developers are working using a centralised code, the end result is consistent regardless of who's working on the project and which Web browser is being used.

Customizable : If you are using only a few features of bootstrap, you can customize to download only those features using the following bootstrap customize page. http://getbootstrap.com/customize/

Support : As Bootstrap is the most popular framework, it has a very large community base and excellent documentation. Bootstrap's excellent documentation, examples and demos helps a developer learn bootstrap quickly even if you are new to it. If you ever run into an issue you will usually get help quickly and easily from the vast online community and Web forums.

Files in "css" folder

bootstrap.css - This is the core css for BootStrap that defines all the style for various controls and components

bootstrap.css.map - When debugging the minified code, the line numbers do not refer to the orignal files. The file that has the .map extension which is also called as source map file fixes this problem by allowing the Web debuggers to refer to the original context from where the code was generated. This file is useful during development.

bootstrap.min.css - This is the compressed version meaning all the whitespaces, line breaks and any other extra characters have been removed. As a result the size of the minified file is smaller than the non-minified file. Minified version is usually used on a production server for efficient download where as the non minified version is used in development environment as it is more readable and easy to debug if there are issues.

bootstrap.min.css.map - Source map file for bootstrap.min.css

bootstrap-theme.css - As the name suggests this is the theme for bootstrap. Adding the core bootstrap.css is enough for bootstrap to work. The theme file is optional and is usually used for a visually enhanced experience. For example if you want 3D effects, gradients, shadows etc.

bootstrap-theme.css.map - Source map file for bootstrap-theme.css

bootstrap-theme.min.css - Minified version of bootstrap-theme.css

bootstrap-theme.min.css.map - Source map file for bootstrap-theme.min.css

Files in "fonts" folder

There are 5 different font files from Glyphicons. These 5 different files are just different format of the Glyphicons font, to support different browsers.

Files in "js" folder : These JavaScript files are optional. These are required if you want to use bootstrap widgets such as picture carousel, dropdown menus, collapsible accordian etc. One important thing to keep in mind is that boostrap JavaScript has a dependency on jQuery, so a reference to jQuery must also exist on the page where you want to use Bootstrap.

bootstrap.js - This is the non-minified readable version that is usually used during development.

bootstrap.min.js - Minified version of bootstrap.js optimised for faster download. This is the version that is usually used in a production environment.

npm.js - npm is a file from Node.js and is used for npm installing bootstrap. If you are new to Node.js, don't worry, this is not going to come in the way to understand bootstrap.

~~~ End of Article ~~~

Session 2: Understanding Programming in Bootstrap

Layout components in Bootstrap

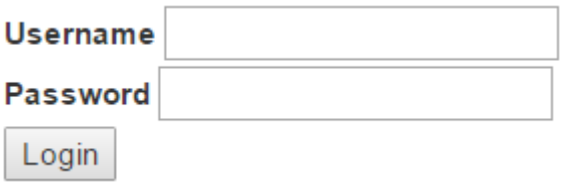
Source	http://csharp-video-tutorials.blogspot.in/2016/06/bootstrap-forms.html
Date of Retrieval	22/03/2017

Bootstrap provides the following 3 form layouts
1. Vertical form (Default)
2. Horizontal form
3. Inline form

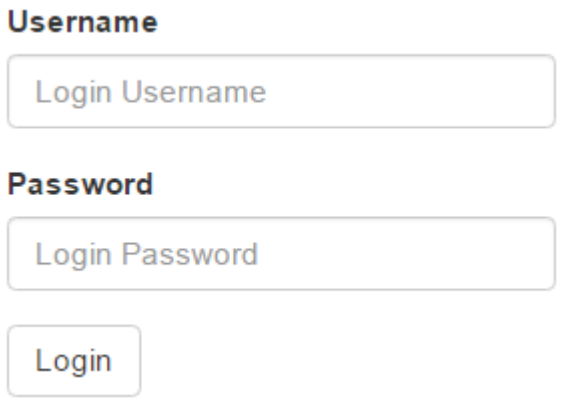
The followin are the classes that are used to style forms
form-group - Use this class on the element that wraps labels and form controls for optimum spacing
form-control - Use this class on all textual elements (<input>, <textarea>, and <select>)

```
<form>
   <div>
      <label for="inputUserName">Username</label>
      <input type="text" id="inputUserName" />
   </div>
   <div>
      <label for="inputPassword">Password</label>
      <input type="password" id="inputPassword" />
   </div>
   <button type="submit">Login</button>
</form>
```

At the moment, we have not applied any of the Bootstrap form classes. If we view this page, in the browser this is how the form looks.

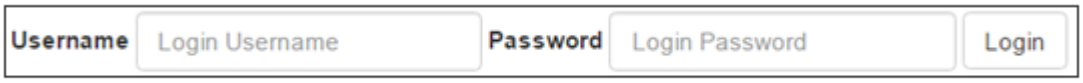


Creating a vertical form layout : This is the default form layout. In the vertical form layout, the label and it's associated form control are stacked. Creating a vertical bootstrap form is very simple.
1. Apply form-group class on the element that wraps the label and the textbox
2. Apply form-control class on the textbox




```
<form>
   <div class="form-group">
      <label for="inputUserName">Username</label>
      <input class="form-control" placeholder="Login Username"
            type="text" id="inputUserName" />
   </div>
   <div class="form-group">
      <label for="inputPassword">Password</label>
      <input class="form-control" placeholder="Login Password"
            type="password" id="inputPassword" />
   </div>
   <button type="submit" class="btn btn-default">Login</button>
</form>
```

Creating an inline form layout : Inline form layout places the form controls side by side. To create an inline form layout, all you have to do is apply form-inline class on the <form> element.



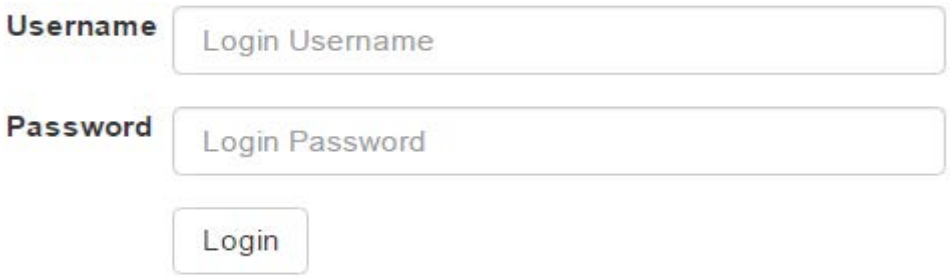

```
<form class="form-inline">
   <div class="form-group">
      <label for="inputUserName">Username</label>
      <input class="form-control" placeholder="Login Username"
            type="text" id="inputUserName" />
   </div>
   <div class="form-group">
      <label for="inputPassword">Password</label>
      <input class="form-control" placeholder="Login Password"
            type="password" id="inputPassword" />
   </div>
   <button type="submit" class="btn btn-default">Login</button>
</form>
```

Please note : You get the inline form layout only if the viewport is at least 768px wide. If

the viewport width falls below 768px, the form layout reverts to vertical.

Creating horizontal form layout : In the horizontal form layout, the label is on the left and it's associated form control is on the right in a single line. To create a horizontal form layout

1. Use the form-horizontal class on the <form> element
2. Use the control-label class on the <label> element
3. Use Bootstrap's grid classes to align labels and form controls




```
<form class="form-horizontal">
  <div class="form-group">
    <label for="inputUserName" class="control-label col-sm-2">Username</label>
    <div class="col-sm-10">
      <input class="form-control" placeholder="Login Username"
          type="text" id="inputUserName" />
    </div>
  </div>
  <div class="form-group">
    <label for="inputPassword" class="control-label col-sm-2">Password</label>
    <div class="col-sm-10">
      <input class="form-control" placeholder="Login Password"
          type="password" id="inputPassword" />
    </div>
  </div>
  <div class="form-group">
    <div class="col-sm-offset-2 col-sm-10">
      <button type="submit" class="btn btn-default">Login</button>
    </div>
  </div>
</form>
```

Please note : With this example, we will only have the horizontal form layout on a large, medium, and small screen sizes. On an extra small screen size, such as a mobile phone, the form reverts to a vertical layout. If you want the horizontal layout on all screen sizes, including an extra small screen size, use col-xs-* grid classes instead of col-sm-* classes.

~~~ End of Article ~~~

Session 3: Basics of jQuery

Understanding concepts of jQuery

Source	http://openmymind.net/2011/3/16/Foundations-of-Programming-2-Appendix-A-jQuery-Bas/
Date of Retrieval	03/04/2017

jQuery Basics

There's a common anecdote given by Web developers, it goes something such as I hated JavaScript, then jQuery came along and now I love it. jQuery is a JavaScript library which takes the pain out of manipulating the DOM and creating reusable JavaScript code. It isn't the only framework of its kind, but it has established itself as the most popular. Part of what makes jQuery so powerful is that it focuses on a few specific things, allowing it to be very good at those. There are two parts to mastering jQuery: first the basics of the library, then how to use the basics to build your own plugins.

A huge part of knowing jQuery is knowing the jQuery method and jQuery object. The jQuery method is responsible for turning a normal HTML DOM element into a jQuery object. $ a shorthand for jQuery, the two are interchangeable, but most people prefer to use $ (I know, it seems magical, but it's just a function name - JavaScript allows such characters in function names). A jQuery object is a wrapper around a DOM element which provides all type of useful manipulation and traversing methods. Let's first look at the jQuery method, and then look at jQuery objects.

In its simplest form, the jQuery method will turn a DOM element into a jQuery element:

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

  <script type="text/javascript">

     var domElement = document.getElementById('main');

     var $main = $(domElement);

     //or (same thing):

     var $main = jQuery(domElement);

  </script>
```

(using $ as a prefix to jQuery object variables is a convention I such as, if you don't, don't use it.)

Being able to turn a DOM element into a jQuery object is useful, as we'll see in a bit. However, the real power of the jQuery method (and jQuery in general) is its ability to take a CSS selector and turn that directly into a jQuery object wrapping the underlying DOM. This is nice for a couple reasons. First, you probably already know how CSS selectors work. Second, CSS selectors have proven to be quite flexible and powerful at selecting elements. Knowing this, we can drop the call to getElementById from the above example and simply do:

```
var $main = $('#main');
```

Because, in CSS, you target an element by id using the #ID selector. Also, jQuery takes care of cross browser incompatibilities for you. It doesn't matter that IE6 (or 7) doesn't support attribute selector such as input[type="text"]. The jQuery selector syntax is such as CSS selectors, but it uses its own engine, independent of the browser's.

jQuery object

Before we dig too deeply into the jQuery method, let's get up to speed with what it returns: a jQuery object. Actually, the method always returns an array of jQuery objects - even when you know it should just return a single value, such as in the above code when we selected by id. This actually turns out to provide some nice consistency, in addition to cleaner code (no null checking), as should become evident as we move forward. A jQuery object can be manipulated via various built-in jQuery methods, 3rd party plugins, or your own custom plugins. For now, we'll stick to the built-in methods. Let's look at some examples:

```
$main.text('welcome!');

$main.addClass('heading');

$main.click(function()

{

   alert('WELCOME!');

});
```

As a general rule, a jQuery method (built-in or not) returns the array of jQuery object being manipulated. This means that method calls are meant to be chained, so the above can be rewritten as:

```
$main.text('welcome').addClass('heading').click(function()

{

   alert('WELCOME!');

});
```

When dealing with an array of jQuery objects with more than 1 element, all elements items are affected.

~~~ End of Article ~~~

Session 4: Functions, Widgets, Plugins in jQuery

Understanding Functions, Widgets, Plugins in jQuery

Source	http://www.pontikis.net/blog/top-10-resources-for-jquery-plugins
Date of Retrieval	5/04/2017

jQuery JavaScript library is very popular among Website developers. According to W3Techs, jQuery adoption is about 55%. It is normal for such a popular library to have many plugins available. Google is always an option when someone tries to find a plugin. Moreover, many plugin repositories or collections are available.

When John Resig developed the jQuery library back in 2006, he can't have imagined that it would become the most popular JavaScript library on the Web, or that it would have tens of thousands of plugins written for it. But it's true to say that jQuery, above all other libraries out there, has been embraced by the Web design community.

It is a fantastic library for designing and developing user interactions quickly. Whether it's an image gallery or form, content-revealing CSS animation or an explosion effect, the library provides the core building-blocks to allow you rapid prototyping and to deliver a unique user interface with the minimum of code and effort.

This presents an interesting question, however. Just because you can roll your own solution to any given problem, does that mean you should? Of course not! There's absolutely no need to reinvent the wheel every time you want to create a bit of common functionality; use plugins to instantly add a behaviour. Doing so will save you even more time and effort!

How to develop a Plug-in

This is very simple to write your own plug-in. Following is the syntax to create a a method –

```
jQuery.fn.methodName = methodDefinition;
```

Here methodNameM is the name of new method and methodDefinition is actual method definition. The guideline recommended by the jQuery team is as follows –

- Any methods or functions you attach must have a semicolon (;) at the end.

- Your method must return the jQuery object, unless explicity noted otherwise.
- You should use this.each to iterate over the current set of matched elements - it produces clean and compatible code that way.
- Prefix the filename with jquery, follow that with the name of the plugin and conclude with .js.
- Always attach the plugin to jQuery directly instead of $, so users can use a custom alias via noConflict() method.

For example, if we write a plugin that we want to name debug, our JavaScript filename for this plugin is –

```
jquery.debug.js
```

The use of the jquery. prefix eliminates any possible name collisions with files intended for use with other libraries.

Example

Following is a small plug-in to have warning method for debugging purpose. Keep this code in jquery.debug.js file –

```
jQuery.fn.warning = function() {
  return this.each(function() {
    alert('Tag Name:"' + $(this).prop("tagName") + '".');
  });
};
```

Here is the example showing usage of warning() method. Assuming we put jquery.debug.js file in same directory of html page.

```
<html>

  <head>


 <script type = "text/javascript"
```

```
      src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js"></script>
    <script src = "jquery.debug.js" type = "text/javascript"></script>
    <script type = "text/javascript" language = "javascript">
      $(document).ready(function() {
        $("div").warning();
        $("p").warning();
      });
    </script>
  </head>
  <body>
    <p>This is para</p>
    <div>This is div</div>
  </body>
</html>
```

Output:

This is para

This is div

~~~ End of Article ~~~

Session 5: Creating Bootstrap Responsive Layout

Bootstrap Responsive Utility Classes

Source	http://csharp-video-tutorials.blogspot.in/2016/05/bootstrap-3-responsive-utility-classes.html
Date of Retrieval	09/04/2017

Bootstrap responsive utility classes are useful for showing and hiding content by device via media query.

The following is the HTML Here is what we want to do
1. The column that displays "ALL Screens" must be visible on all screen sizes
2. The column that displays "Medium, Large and Small" must be visible only on medium, large and small screen sizes. It should be hidden on an extra small screen size.
3. The column that displays "Medium and Large" must be visible only on medium and large screen sizes. It should be hidden on small and extra small screen sizes.
4. The column that displays "Large" must be visible only on a large screen size. It should be hidden on all other screen sizes.

The following is the HTML I used to produce these 4 columns

```
<div class="container">
  <div class="row">
    <div class="col-lg-3 col-md-4 col-sm-6 col-xs-12">
      <div class="customDiv">ALL Screens</div>
    </div>
    <div class="col-lg-3 col-md-4 col-sm-6 col-xs-12">
      <div class="customDiv">Medium, Large and Small</div>
    </div>
    <div class="col-lg-3 col-md-4 col-sm-6 col-xs-12">
      <div class="customDiv">Medium and Large </div>
    </div>
    <div class="col-lg-3 col-md-4 col-sm-6 col-xs-12">
      <div class="customDiv">Large</div>
    </div>
  </div>
</div>
```

This can be very easily achieved by using the responsive utility classes
1. "ALL Screens" column must be visible on all screen sizes, so no change is required here.

2. "Medium, Large and Small" column must be visible only on medium, large and small screen sizes. It should be hidden on an extra small screen size. So applying "hidden-xs" class on this column will hide this column on an extra small device but will be visible across all other devices.

```
<div class="col-lg-3 col-md-4 col-sm-6 col-xs-12 hidden-xs">
   <div class="customDiv">Medium, Large and Small</div>
</div>
```

We can also achieve exactly the same thing by using visible-lg, visible-md, and visible-sm classes instead of hidden-xs class. Since with "hidden-xs" we only need to use one class, where as with visible classes we have to use 3 of them, so I have chosen to use hidden-xs.

```
<div class="col-lg-3 col-md-4 col-sm-6 col-xs-12 visible-lg visible-md visible-sm">
   <div class="customDiv">Medium, Large and Small</div>
</div>
```

3. "Medium and Large" column must be visible only on medium and large screen sizes. It should be hidden on small and extra small screen sizes. To achieve this apply visible-lg and visible-md classes on this column.

```
<div class="col-lg-3 col-md-4 col-sm-6 col-xs-12 visible-lg visible-md">
   <div class="customDiv">Medium and Large </div>
</div>
```

We can also achieve exactly the same thing by using hidden-sm and hidden-xs classes instead of visible-lg and visible-md classes.

```
<div class="col-lg-3 col-md-4 col-sm-6 col-xs-12 hidden-sm hidden-xs">
   <div class="customDiv">Medium and Large </div>
</div>
```

4. "Large" column must be visible only on large screen sizes. It should be hidden on all other screen sizes. To achieve this apply visible-lg class on this column.

```
<div class="col-lg-3 col-md-4 col-sm-6 col-xs-12 visible-lg">
   <div class="customDiv">Large</div>
</div>
```

As you might have already guessed by now, you can also achieve exactly the same thing by using the hidden classes (hidden-xs hidden-sm and hidden-md), but with this approach you have to use 3 classes, where as with visible classes we have to use only one class (visible-lg)

```
<div class="col-lg-3 col-md-4 col-sm-6 col-xs-12 hidden-xs hidden-sm hidden-md">
   <div class="customDiv">Large</div>
</div>
```

used to produce these 4 columns

```
<div class="container">
   <div class="row">
      <div class="col-lg-3 col-md-4 col-sm-6 col-xs-12">
         <div class="customDiv">ALL Screens</div>
      </div>
      <div class="col-lg-3 col-md-4 col-sm-6 col-xs-12">
         <div class="customDiv">Medium, Large and Small</div>
      </div>
      <div class="col-lg-3 col-md-4 col-sm-6 col-xs-12">
         <div class="customDiv">Medium and Large </div>
      </div>
      <div class="col-lg-3 col-md-4 col-sm-6 col-xs-12">
         <div class="customDiv">Large</div>
      </div>
   </div>
</div>
```

table class provides light padding and horizontal lines

```
<table class="table">
   <!--rest of the HTML stays the same-->
</table>
```

table-striped class provides zebra-striping for the table rows

```
<table class="table table-striped">
   <!--rest of the HTML stays the same-->
</table>
```

table-bordered class provide borders on all sides of the table and cells.

```
<table class="table table-striped table-bordered">
   <!--rest of the HTML stays the same-->
</table>
```

table-hover class provides highlighting of rows on hover

```
<table class="table table-bordered table-hover">
   <!--rest of the HTML stays the same-->
</table>
```

table-condensed class makes table more compact by cutting cell padding in half

```
<table class="table table-bordered table-hover table-condensed">
   <!--rest of the HTML stays the same-->
</table>
```

To make a table responsive, place the table inside a element, and apply table-responsive class on the element. This will provide a horizontal scrollbar when the screen size is less than 768px (i.e on a small device**). On a screen siz**e larger than 768px you will not find any difference. Applying the table-responsive class directly on the table will not do anything useful.

```
<div class="table-responsive">
   <table class="table table-bordered table-hover">
      <!--rest of the HTML stays the same-->
   </table>
</div>
```

~~~ End of Article ~~~

Session 6: Creating Responsive Webpages using Bootstrap and jQuery

Customizing Bootstrap

Source	http://blog.neverendingo.de/customizing-bootstrap/
Date of Retrieval	09/04/2017

Most people doing Websites from time to time should already know the wonderful CSS Framework Bootstrap by now. Neverland, the theme project we use on a lot of KDE sites these days is also based on it, and i really such as it a lot. It makes use of Less CSS, which makes editing and updating the CSS a charm.
Actually, i find it quite often when surfing the Web. And that is the point of this post. I do notice it quite often, as many people seem to just put it on their Website without any adjustments, just as is. And to draw the connection to KDE, i have noticed that on some small KDE app sites as well.

But why not give it a unique look and still benefit from the great predefined classes bootstrap offers? Shouldn't take a lot of time. Let's have a look.

Prerequisites

First, as i already said, Bootstrap uses several less files which finally get "compiled" into a single CSS file. And that is my personal favourite for working with Bootstrap. So i would recommend going to their github page and cloning the repo, as the download doesn't have the corresponding less files, just the final CSS (and Javascript, if wished). If you cloned it, you will also get a folder called "less". That is where the magic happens.

But another thing you would need is either the less.js file (for client side rendering of less files) or the less module for node.js. I won't digg into this, just follow the links above to get an idea what fits best in your situation. For the sake of this tutorial i will use the former method and use client side rendering as long as we are going through it.

Planning

Of course we need a rough idea on what we want to reach. In this case i am thinking of what comes to my mind when i usually read tutorials. Coffee. Coffee and newspapers (or newsreaders). This means some typewriter such as font, some brown, milky and smooth colors. We keep that in mind for later.

Let's get started

We take one of the Bootstrap example files to work with. I decided to use “marketing-narrow.html” from the docs/examples folder. The default look is such as this:
Back to our less files from Bootstrap. The most important files are bootstrap.less and variables.less. The former is responsible for loading all the single less files into the final output. The latter holds the most common variables for the entire look of Bootstrap and should be your first stop to get used to Bootstrap.
And right on top we have some common color codes we could adjust. The defaults are:

```
// Grays

// -------------------------

@black:                 #000;

@grayDarker:            #222;

@grayDark:              #333;

@gray:                  #555;

@grayLight:             #999;

@grayLighter:           #eee;

@white:                 #fff;




// Accent colors

// -------------------------

@blue:                  #049cdb;

@blueDark:              #0064cd;
```

```
@green:                 #46a546;

@red:                   #9d261d;

@yellow:                #ffc40d;

@orange:                #f89406;

@pink:                  #c3325f;

@purple:                #7a43b6;
```

Now remembering our initial idea to have something newspaper such as, i think we should make the white a bit less white, and the black a bit less black and maybe add some yellow to it. That means changing @white to something such as #f4f4eb and @black to something such as #232322.
Note: This is just an example, use whatever values you prefer
A generic customisation of the grays could be:

```
@black:                 #232322;

@grayDarker:            #3c3c39;

@grayDark:              #42423f;

@gray:                  #60605c;

@grayLight:             #83837e;

@grayLighter:           #c8c8c0;

@white:                 #f4f4eb;
```

These values will be used across most of the Bootstrap components already. But before previewing that, let's also adjust the font for the headings. I such as the use of font loading, so let's see if we can find a good font online.

For including a Webfont we also need to adjust bootstrap.less, and we use the @import method here. The top of bootstrap.less should now look such as:

```
@import url("http://fonts.googleapis.com/css?family=Cinzel");
```

```
// Core variables and mixins

@import "variables.less"; // Modify this for custom colors, font-sizes, etc

@import "mixins.less";
```

Now we also need to make sure headings will use it. Back in variables.less we change this section

```
@headingsFontFamily:    'Cinzel', serif;

@headingsFontWeight:    normal;

@headingsColor:                inherit; // empty to use BS default, @textColor
```

That's enough to change all headings on every subsequent page we will publish. How does it look at this point?Looks such as we are getting there, no? Already looks more unique. And we only altered 10 lines so far.
Now let's see what we can do with the builtin mixins of Bootstrap. Looking through the mixins.less file we notice

```
// Drop shadows

.box-shadow(@shadow) {

-Webkit-box-shadow: @shadow;

-moz-box-shadow: @shadow;

box-shadow: @shadow;

}
```

Using this we could easily add drop shadows to elements with one calls that outputs all the necessary vendor prefixes.I think the buttons would look nice with a drop shadow, don't you think? They already have one, but not such as what i prefer. Opening up buttons.less we can change

```
.btn {

...
```

```
.box-shadow(~"inset 0 1px 0 rgba(255,255,255,.2), 0 1px 2px rgba(0,0,0,.05)");

...
```

to something such as

```
.btn {

...

.box-shadow(0px 2px 3px 1px fade(@black, 50%));

...
```

This quick change shows also the magic of editing less files instead of plain CSS. We replaced a hardcoded color value with the global variable @black, but not only that. We also run the lesscss function fade() on it, which returns a rgba value of the color you provided.

Now, every time we think the black color doesn't satisfy our needs, we change it in one place and all other references get updated as well. Hardly possible with simple CSS.
So, let's have a last lookIt is up to you to decide if this looks good (probably not). But the point was to show how simple it is to get a unique look while using a CSS Framework such as Bootstrap. The limits are up to you.
To summarize: 11 lines changed, less than half an hour, new custom look.

~~~ End of Article ~~~