

mEm Inc

SECOND EDITION

Bootstrap

Reference Guide

WEB DEVELOPMENT WITH BOOTSTRAP

CLAUDIA ALVES & ALEXANDER ARONOWITZ



Bootstrap Reference Guide

Web Development with Bootstrap

2nd edition

2020

By Claudia Alves

"Programming isn't about what you know; it's about what you can figure out." - *Chris Pine*

INTRODUCTION

WHAT IS BOOTSTRAP?

BOOTSTRAP FEATURES

CREATE AN HTML PAGE

LOAD BOOTSTRAP VIA CDN

HOST BOOTSTRAP LOCALLY

LOAD BOOTSTRAP JAVASCRIPT

PUT IT ALL TOGETHER

CHAPTER 1

DESIGNING WITH GRID

PREPARING THE PAGE

HTML5 DOCTYPE IS REQUIRED

THE MOBILE IS THE MOST IMPORTANT

RESPONSIVE IMAGES

TYPOGRAPHY AND LINKS

STYLE NORMALIZATION

CENTERING THE CONTENTS OF THE PAGE

INTRODUCTION

NOTE

MEDIA QUERIES

EXAMPLE OF GRID CREATED WITH BOOTSTRAP

VARIABLE WIDTH CONTAINER EXAMPLE

RESETTING COLUMNS

MOVING COLUMNS

NESTING COLUMNS

REARRANGING THE COLUMNS

LESS VARIABLES AND MIXINS

VARIABLES

MIXINS

EXAMPLE OF USE

TYPOGRAPHY

HEADLINES

TEXTS

TEXT DESTACADO

LESS VARIABLES

EMPHASIS

UNIMPORTANT TEXT

BOLD TEXT

ALTERNATIVE LABELS

CSS CLASSES

CSS CLASSES TO ALIGN TEXT

CSS INDICAT CLASES PARA EL TIPO DE CONTENIDO

ABBREVIATIONS

BASIC ABBREVIATIONS

INITIALS

ADDRESSES

BLOCKQUOTES

BLOCKQUOTE BY DEFAULT

OPTIONS FOR BLOCKQUOTE ELEMENTS

SHOWING THE SOURCE

MODIFYING THE ALIGNMENT

LISTS

LIST NO ORDENADAS

ORDERED LISTS

LISTS WITHOUT STYLE

ONLINE LISTS

DEFINITION LISTS

HORIZONTAL DEFINITION LISTS

CSS ELEMENTS

BASIC TABLES

SHIELDED TABLES

TABLES WITH BORDERS

DYNAMIC TABLES

CONDENSED TABLES

SEMANTIC TABLES

RESPONSIVE TABLES

IMAGES

FLOATING ELEMENTS

[CENTERED ELEMENTS](#)

[CLEANING FLOATS](#)

[HIDING AND SHOWING ELEMENTS](#)

[HIDDEN CONTENT](#)

[REPLACING TEXT WITH IMAGES](#)

[RESPONSIVE UTILITIES](#)

[FORMS](#)

[BASIC FORM](#)

[ONLINE FORM](#)

[HORIZONTAL FORMS](#)

[FORM FIELDS](#)

[INPUTS](#)

[TEXTAREA](#)

[CHECKBOXES AND RADIO BUTTONS](#)

[CHECKBOXES AND RADIO BUTTONS ONLINE](#)

[DROP-DOWN LISTS](#)

[STATIC FORM FIELDS](#)

[FORM STATES](#)

[SELECTED FIELDS](#)

[FIELDS DISABLED](#)

[VALIDATION STATES](#)

[RESIZING FORM FIELDS](#)

[CHANGING HEIGHT](#)

[BUTTONS](#)

[DIFFERENT SIZE BUTTONS](#)

[ACTIVATED BUTTONS](#)

[ACTIVATING <BUTTON> ELEMENTS](#)

[BUTTON LABELS](#)

[CHAPTER II](#)

[PANELS, WINDOWS MODAL, AND CAROUSEL ELEMENTS](#)

[PANELS PAINTINGS](#)

[HEADER AND FOOTER PANEL](#)

FORMAT THE PANEL ACCORDING TO ITS CONTENTS

ADD TABLES, MENUS, AND OTHER ITEMS TO THE PANEL

MODAL WINDOW

MODERATE WINDOW SIZES

COLLAPSE

SHRINK THE CONTENTS OF OTHER ELEMENTS

CAROUSEL REPEATING DISPLAY TOOL

CONTENTS OF THE REPEATING VIEWER TOOL

CHAPTER III

DROPDOWNS, TABS, AND NAVBARS

DROP DOWNS

ADD A TITLE AND COMMAS BETWEEN PARTS OF THE LIST

ALIGN THE LIST

TABS

NAVBAR MENU BAR

RESPONSIVE MENU BAR

ADD OTHER ITEMS IN THE MENU BAR

FIXED MOBILE MENU BAR

STATIC MENU BAR

MORE FORMATS AND ITEMS IN THE LIST

CHAPTER IV

THE TREE MENU

CHAPTER V

BUILD A SITE

USE BOOTSTRAP WITHOUT LESS

USE BOOTSTRAP WITH LESS

BUILD THE PROJECT USING BOOTSTRAP

COMPONENTS

[BASIC FORMAT](#)

[HTML STRUCTURE](#)

[PAGE LAYOUT](#)

[CONCLUSION](#)

[CHAPTER VI](#)

[BUILD A SITE](#)

[LOGO](#)

[NAVIGATION BAR](#)

[PAGE HEADER](#)

[SUBMENU](#)

[SIDEBAR](#)

[MAIN CONTENT](#)

[STAFF SECTION](#)

[TWITTER FEED](#)

[SITEMAP](#)

[SOCIAL ICONS](#)

[COPYRIGHT TEXT](#)

Introduction

Bootstrap is a front-end framework that helps you build mobile responsive websites more quickly and easily. First developed by Twitter, Bootstrap is by now used for anything from developing web applications to WordPress themes. It is also completely free, versatile, and intuitive.

With Bootstrap, you can conjure complex web pages from standard HTML and customize them to your needs. Bootstrap also comes with a number of jQuery plugins that can provide additional functionality such as carousels, buttons, tooltips, and more.

Last, but not least, it gives you a lot of shortcuts for creating web pages that will save you time and energy. All you need is a basic understanding of HTML and CSS to create ones that are responsive, mobile-first, and compatible with all modern browsers.

What is Bootstrap?

Bootstrap is a responsive HTML / CSS framework. Bootstrap contains the basic tools and elements you will need in most of your projects such as: tables, buttons, text and form tools, and secondary: heading, navigation, backgrounds, and more. Bootstrap saves designers a lot of time and effort while designing pages. All they need to know is the class names they will work with to give the appropriate formatting to their element. In addition, the Bootstrap framework makes the design fit and responsive to all different screen sizes.

Bootstrap features

The advantages of the Bootstrap framework that contributed to its spread:

- To preserve time and effort, the pre-designed design in this framework has all that deafness has to do is put the elements in the right place and customize them according to the purpose.
- Responsive design and compatible with all browsers, this feature makes the site that you create appear conveniently for all different devices in their sizes and the same format and appearance that you put, in addition to the compatibility of CSS properties with different browsers, which means that no part of the design with the browsers, is considered This is another saving on your effort to try out the design on browsers.
- Easy to use and Consistent, any person with some basics in HTML and CSS can use this framework easily, you may not have to write a single CSS code, and also provides the feature of consistent formatting in all parts of the site, working on a template prepared for this purpose will make the site consistent In the appearance of the components in particular and in the layout of the site in general.

In addition, the Bootstrap framework is open source and you can view its components.

.

Create an HTML page

As a first step, we will create a simple HTML template as a base where we will use Bootstrap. For that, the first thing you want to do is create a folder on your computer or server for the project files. In this case, I will simply call it bootstrap. Here, create a new text file and call it index.html. Open it with a text editor of your choice (e.g. Notepad++) and then paste the code below into it.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Bootstrap Tutorial Sample Page</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
</head>
<body>
</body>
</html>
```

Don't forget to save your file before moving on!

Load Bootstrap via CDN

As already explained, Bootstrap consists mainly of style sheets and scripts. As such, they can be loaded in the header and footer of your web page like other assets such as custom fonts. The framework offers a CDN (content delivery network) access path for that.

To get Bootstrap into your page, simply paste the code below into the `<head>` section of your template.

```
<link rel="stylesheet"  
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"  
integrity="sha384-  
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"  
crossorigin="anonymous">
```

When you now save the file, any browser that opens it will automatically load the Bootstrap assets.

Using the remote method is a good idea as many users will already have the framework in the cache of their browser. If that is the case, they won't have to reload it when coming to your site, leading to faster page loading time. As a consequence, this is the recommended method for live sites.

However, for experimenting and development, or if you want to be independent of an Internet connection, you can also get your own copy of Bootstrap. This is what I am doing for this tutorial because it also results in less code to post.

Host Bootstrap locally

An alternative way to set up Bootstrap is to download it to your hard drive and use the files locally. You find download options in the same place as the links to the remote version. Here, be sure to get the compiled CSS and JS files. You don't need the source files.

Once you have done so, unzip the file and copy its contents into the same directory as index.html. After that, you can load the Bootstrap CSS into your project like this:

```
<link rel="stylesheet" href="bootstrap/css/bootstrap.min.css">
```

You will notice that this includes the file path at which to find the Bootstrap file. In your case, make sure your path corresponds to your actual setup. For example, the names of the directories might differ if you downloaded a different version of Bootstrap.

Load Bootstrap JavaScript

The last step in setting up Bootstrap is to load the Bootstrap JavaScript library. These are included in the downloaded version of the framework and you also find links to remote sources in the same place as before. However, we will load it in a different place than the style sheet. Instead of the header, it goes into the page footer, right after the call for jQuery.

You can call it remotely like this:

```
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"
integrity="sha384-
ChfqquxZUCnJSK3+MXmPNlyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy"
crossorigin="anonymous"></script>
```

Or locally like so:

```
<script src="bootstrap/js/bootstrap.min.js"></script>
```

Put it all together

If you have followed the steps above correctly, you should end up with a file that looks like this for the remote solution:

```
<!DOCTYPE html>
<html lang="en">
  <head>

    <title>Bootstrap Tutorial Sample Page</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
crossorigin="anonymous">
  </head>
  <body>
    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"
integrity="sha384-
ChfqquxZUCnJSK3+MXmPNlyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy"
crossorigin="anonymous"></script>
  </body>
</html>
```

Alternatively, if you are hosting locally, your page template should resemble the code below:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Bootstrap Tutorial Sample Page</title>
```

```
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="bootstrap/css/bootstrap.min.css">
</head>
<body>
<script src="jquery-3.3.1.min.js"></script>
<script src="bootstrap/js/bootstrap.min.js"></script>
</body>
</html>
```

If that is what you have and you saved your work, you are now ready to move on to the next step.

Chapter 1

Designing with grid

Preparing the page

Before starting to design the layout or content structure of the pages, it is necessary to make some important preparations.

HTML5 doctype is required

Bootstrap uses some HTML elements and some CSS properties that require the use of the HTML5 doctype. Don't forget to include this doctype on all your pages with the following code:

```
<!DOCTYPE html>
```

```
<html lang = "is">
```

```
...
```

```
</html>
```

The mobile is the most important

Bootstrap 2 included some utilities to make the pages adapt to mobile devices. Bootstrap 3 was created from scratch with mobile in mind. So instead of including some optional mobile styles, all of that is already included in Bootstrap itself. That's why we like to say that for Bootstrap 3, mobile devices are the most important thing.

In order for pages to display correctly and zoom works well on mobile devices, it is important that you add the following tag inside the <head> header of the pages:

```
<meta name = "viewport" content = "width = device-width, initial-scale = 1">
```

If you want to disable zooming on your pages, add the user-scalable = no property to the previous tag:

```
<meta name = "viewport" content = "width = device-width, initial-scale = 1, maximum-scale = 1, user-scalable = no">
```

By adding the user-scalable = no property, users will no longer be able to zoom in on the page and will only be able to scroll through its contents. The result is that the behavior of the page is more like that of a native mobile app. In any case, limiting the freedoms of users can be counterproductive and therefore, we do not recommend that you use this option on all your sites.

Responsive images

Bootstrap 3 no longer automatically resizes images as it did in Bootstrap 2. To maintain the same behavior as before, you must add the `.img-responsive` class to each image you want to be responsive. This class includes the max-width properties: `100%;` y `height: auto;` so that the image scales according to the size of the element in which it is located.

```
<img src = "..." class = "img-responsive" alt = "Responsive image">
```


Typography and links

Bootstrap sets a series of default styles for the typography of all elements and for the links on the page. Specific:

The body background color is set to white with the background-color: white property;

The value of the @ font-family-base, @ font-size-base and @ line-height-base variables defined by LESS are used as attributes of the typographic properties of the elements.

The color of the links is set to the value of the variable @ link-color of LESS and only the underlined links are shown in the state: hover

This first style initialization is defined in the scaffolding.less file.

Style normalization

To homogenize the initial styles in the different browsers, Bootstrap uses the Normalize style sheet, which is a project created by Nicolas Gallagher and Jonathan Neal.

Centering the contents of the page

If you want to center a page with respect to the browser window, enclose its contents inside an element and apply the `.container` class:

```
<div class = "container">
```

```
...
```

```
</div>
```

Container width varies at each design breakpoint to fit the rack. Containers cannot be nested due to the padding property and their fixed width.

Grating types

Bootstrap includes a fluid grid or grid designed for mobile and compliant with responsive web design. This grid grows to 12 columns as the size of the device screen grows. Bootstrap includes CSS classes to use the grid directly in your layouts and also defines LESS mixins so you can create more semantic layouts.

Introduction

Grid-based page design is done using rows and columns where content is placed. Here's how the Bootstrap grid works:

Rows are always defined within a container of type `.container` (fixed width) or type `.container-fluid` (variable width). In this way the rows are well aligned and show the correct padding.

Rows are used to horizontally group multiple columns

The content is always placed inside the columns, since the rows should only contain elements of the column type as children.

Bootstrap defines many CSS classes (such as `.row` and `.col-xs-4`) to quickly create grids. There are also Mixins of Less to create more semantic designs.

The separation between columns is done by applying padding. To counteract its effects on the first and last columns, the rows (`.row` elements) apply negative margins.

The columns of the grid define their width by specifying how many of the 12 columns in the row they occupy. If for example you want to divide a row into three equal columns, you would use the class `.col-xs-4` (the 4 indicates that each column occupies 4 of the 12 columns into which each row is divided).

Note

If you want to create a completely seamless design that spans the full width of the browser, you should enclose the grids within an element to which you apply the padding styles: 0 15px ;. In this way the margin margins can be neutralized: 0 -15px; that apply to .row elements.

Media queries

Bootstrap uses the following media queries to set the different breakpoints at which the grid transforms to suit each device.

```
/* Very small devices (phones up to 768px wide) */
```

```
/* No media query is defined because this is the style by  
defect used by Bootstrap 3 */
```

```
/* Small devices (tablets, width greater than or equal to 768px) */
```

```
@media (min-width: @ screen-sm-min) {...}
```

```
/* Medium devices (computers, width greater than or equal to 992px) */
```

```
@media (min-width: @ screen-md-min) {...}
```

```
/* Large devices (computers, width greater than or equal to 1200px) */
```

```
@media (min-width: @ screen-lg-min) {...}
```

Sometimes the following media queries are also used that define the max-width property and allow restricting the devices to which the CSS styles are applied:

```
@media (max-width: @ screen-xs-max) {...}
```

```
@media (min-width: @ screen-sm-min) and (max-width: @ screen-sm-max) {...}
```

```
@media (min-width: @ screen-md-min) and (max-width: @ screen-md-max) {...}
```

```
@media (min-width: @ screen-lg-min) {...}
```

Example of grid created with Bootstrap

The following example shows how to create a grid with the `.col-md-*` classes. On mobile devices (extra small or small) this grid is displayed vertically, but on a computer (medium or large) it is viewed horizontally.

```
<div class = "row">
  <div class = "col-md-1">. col-md-1 </div>
  <div class = "col-md-1">. col-md-1 </div>
  <div class = "col-md-1">. col-md-1 </div>
  <div class = "col-md-1">. col-md-1 </div>
  <div class = "col-md-1">. col-md-1 </div>
  <div class = "col-md-1">. col-md-1 </div>
  <div class = "col-md-1">. col-md-1 </div>
  <div class = "col-md-1">. col-md-1 </div>
  <div class = "col-md-1">. col-md-1 </div>
  <div class = "col-md-1">. col-md-1 </div>
  <div class = "col-md-1">. col-md-1 </div>
</div>
<div class = "row">
  <div class = "col-md-8">. col-md-8 </div>
  <div class = "col-md-4">. col-md-4 </div>
</div>
<div class = "row">
  <div class = "col-md-4">. col-md-4 </div>
  <div class = "col-md-4">. col-md-4 </div>
  <div class = "col-md-4">. col-md-4 </div>
</div>
<div class = "row">
  <div class = "col-md-6">. col-md-6 </div>
```

```
<div class = "col-md-6">. col-md-6 </div>  
</div>
```


Variable width container example

If you want to transform a fixed-width grid into a variable-width grid that spans the entire width of the browser, replace the CSS `.container` class with `.container-fluid` on the element that encloses all other grid elements:

```
<div class = "container-fluid">  
  <div class = "row">  
    ...  
  </div>  
</div>
```

Resetting columns

Since Bootstrap grids have four breakpoints at which columns are rearranged, you are almost certainly going to run into trouble when columns are of different heights. To fix it, use the `.clearfix` class combining it with one of the `.visible-xs` auxiliary classes:

```
<div class = "row">  
  <div class = "col-xs-6 col-sm-3">. col-xs-6 .col-sm-3 </div>  
  <div class = "col-xs-6 col-sm-3">. col-xs-6 .col-sm-3 </div>
```

<! - The 'clearfix' class only applies when the device is
very small, as indicated by the class 'visible-xs' ->

```
<div class = "clearfix visible-xs"> </div>  
  
<div class = "col-xs-6 col-sm-3">. col-xs-6 .col-sm-3 </div>  
<div class = "col-xs-6 col-sm-3">. col-xs-6 .col-sm-3 </div>  
</div>
```

It is also possible that sometimes you need to reset the displacements of the columns. Classes that reset these values are only available for medium and large devices, that offsets only work on those devices.

```
<div class = "row">  
  <div class = "col-sm-5 col-md-6">. col-sm-5 .col-md-6 </div>  
  <div class = "col-sm-5 col-sm-offset-2 col-md-6 col-md-offset-0">. col-sm-5 .col-sm-offset-2  
  .col-md-6 .col-md-offset-0 </div>  
</div>
```

```
<div class = "row">
```

<div class = "col-sm-6 col-md-5 col-lg-6">. col-sm-6 .col-md-5 .col-lg-6 </div>

<div class = "col-sm-6 col-md-5 col-md-offset-2 col-lg-6 col-lg-offset-0">. col-sm-6 .col-md-5
.col -md-offset-2 .col-lg-6 .col-lg-offset-0 </div>

</div>

Moving columns

Add the `.col-md-offset-*` class to shift any column to its right. These classes increase the size of the left margin of the column by an amount equivalent to those * columns. The `.col-md-offset-4` class for example shifts the column a width equivalent to 4 columns.

```
<div class = "row">  
  <div class = "col-md-4">. col-md-4 </div>  
  <div class = "col-md-4 col-md-offset-4">. col-md-4 .col-md-offset-4 </div>  
</div>
```

```
<div class = "row">  
  <div class = "col-md-3 col-md-offset-3">. col-md-3 .col-md-offset-3 </div>  
  <div class = "col-md-3 col-md-offset-3">. col-md-3 .col-md-offset-3 </div>  
</div>
```

```
<div class = "row">  
  <div class = "col-md-6 col-md-offset-3">. col-md-6 .col-md-offset-3 </div>  
</div>
```

Nesting columns

Bootstrap 3 also allows nesting of columns within other columns. To do this, inside a column with class `col-md-*` create a new element with class `.row` and add one or more columns with class `.col-md-*`. Nested columns always have to add 12 columns in width, as the following example shows.

```
<div class = "row">  
  <div class = "col-md-9">  
    Level 1: .col-md-9  
    <div class = "row">  
      <div class = "col-md-6">  
        Level 2: .col-md-6  
      </div>  
      <div class = "col-md-6">  
        Level 2: .col-md-6  
      </div>  
    </div>  
  </div>  
</div>
```

Rearranging the columns

Bootstrap 3 introduces the ability to reorder columns to change their position, which is very important for responsive web designs. Add the `.col-md-push-*` and `.col-md-pull-*` classes to reorder the columns.

```
<div class = "row">  
  <div class = "col-md-9 col-md-push-3">. col-md-9 .col-md-push-3 </div>  
  <div class = "col-md-3 col-md-pull-9">. col-md-3 .col-md-pull-9 </div>  
</div>
```

LESS variables and mixins

In addition to the ready-made CSS classes for quickly defining grids, Bootstrap includes LESS variables and mixins to easily generate your own semantic web designs.

Variables

Variables set the number of columns, their spacing, and the width of the browser from which the columns float horizontally instead of being displayed on top of each other. The default values of these variables are as shown below:

@ grid-columns: 12;

@ grid-gutter-width: 30px;

@ grid-float-breakpoint: 768px;

Mixins

The mixins, together with the previous variables, allow creating semantic styles for the different elements of the grid.

```
// Create a multi-column container element
.make-row (@gutter: @ grid-gutter-width) {
  // Clean up the floated columns
  .clearfix ();

  @media (min-width: @ screen-small) {
    margin-left: (@gutter / -2);
    margin-right: (@gutter / -2);
  }

  // Apply a negative margin to the row to align the
  // column content
  .row {
    margin-left: (@gutter / -2);
    margin-right: (@gutter / -2);
  }
}

// Generate the extra small columns
.make-xs-column (@columns; @gutter: @ grid-gutter-width) {
  position: relative;

  // Prevent columns from being seen when empty
  min-height: 1px;

  // Use padding to separate the columns
  padding-left: (@gutter / 2);
  padding-right: (@gutter / 2);
}
```

```
// Calculate the width based on the number of columns
@media (min-width: @ grid-float-breakpoint) {
    float: left;
    width: percentage (((@ columns / @ grid-columns)));
}
}
```

```
// Generate the small columns
.make-sm-column (@columns; @gutter: @ grid-gutter-width) {
    position: relative;
    // Prevent columns from being seen when empty
    min-height: 1px;
    // Use padding to separate the columns
    padding-left: (@gutter / 2);
    padding-right: (@gutter / 2);
}
```

```
// Calculate the width based on the number of columns
@media (min-width: @ screen-small) {
    float: left;
    width: percentage (((@ columns / @ grid-columns)));
}
}
```

```
// Generate the small column offsets
.make-sm-column-offset (@columns) {
    @media (min-width: @ screen-small) {
        margin-left: percentage (((@ columns / @ grid-columns)));
    }
}

.make-sm-column-push (@columns) {
```

```

    @media (min-width: @ screen-small) {
        left: percentage ((@ columns / @ grid-columns));
    }
}

.make-sm-column-pull (@columns) {
    @media (min-width: @ screen-small) {
        right: percentage ((@ columns / @ grid-columns));
    }
}

// Generate the medium columns
.make-md-column (@columns; @gutter: @ grid-gutter-width) {
    position: relative;
    // Prevent columns from being seen when empty
    min-height: 1px;
    // Use padding to separate the columns
    padding-left: (@gutter / 2);
    padding-right: (@gutter / 2);

    // Calculate the width based on the number of columns
    @media (min-width: @ screen-medium) {
        float: left;
        width: percentage ((@ columns / @ grid-columns));
    }
}

// Generate the displacements of the medium columns
.make-md-column-offset (@columns) {
    @media (min-width: @ screen-medium) {
        margin-left: percentage ((@ columns / @ grid-columns));
    }
}

```

```

}
.make-md-column-push (@columns) {
  @media (min-width: @ screen-medium) {
    left: percentage ((@ columns / @ grid-columns));
  }
}

.make-md-column-pull (@columns) {
  @media (min-width: @ screen-medium) {
    right: percentage ((@ columns / @ grid-columns));
  }
}

// Generate the large columns
.make-lg-column (@columns; @gutter: @ grid-gutter-width) {
  position: relative;
  // Prevent columns from being seen when empty
  min-height: 1px;
  // Use padding to separate the columns
  padding-left: (@gutter / 2);
  padding-right: (@gutter / 2);

  // Calculate the width based on the number of columns
  @media (min-width: @ screen-large) {
    float: left;
    width: percentage ((@ columns / @ grid-columns));
  }
}

// Generate offsets for large columns
.make-lg-column-offset (@columns) {
  @media (min-width: @ screen-large) {

```

```
    margin-left: percentage ((@ columns / @ grid-columns));
  }
}

.make-lg-column-push (@columns) {
  @media (min-width: @ screen-large) {
    left: percentage ((@ columns / @ grid-columns));
  }
}

.make-lg-column-pull (@columns) {
  @media (min-width: @ screen-large) {
    right: percentage ((@ columns / @ grid-columns));
  }
}
```

Example of use

Using the previous mixins and modifying the value of the variables to adjust them to your needs, you can now create semantic web designs. This example shows how to create a two-column layout with a separation between the two:

```
.wrapper {  
  .make-row ();  
}  
  
.content-main {  
  .make-column (8);  
}  
  
.content-secondary {  
  .make-column (3);  
  .make-column-offset (1);  
}  
  
<div class = "wrapper">  
  <div class = "content-main"> ... </div>  
  <div class = "content-secondary"> ... </div>  
</div>
```

Typography

Styles related to typography and content text are essential in any CSS framework. For that reason, Bootstrap 3 includes dozens of styles for the main elements used in sites and web applications.

Headlines

Bootstrap 3 defines default styles for all levels of page headlines, from `<h1>` to `<h6>`. Example:

`<h1>` **h1. Bootstrap heading** `</h1>`

`<h2>` **h2. Bootstrap heading** `</h2>`

`<h3>` **h3. Bootstrap heading** `</h3>`

`<h4>` **h4. Bootstrap heading** `</h4>`

`<h5>` **h5. Bootstrap heading** `</h5>`

`<h6>` **h6. Bootstrap heading** `</h6>`

This is what this example looks like in your browser:

<h1>h1. Bootstrap heading</h1>	Semibold 36px
<h2>h2. Bootstrap heading</h2>	Semibold 30px
<h3>h3. Bootstrap heading</h3>	Semibold 24px
<h4>h4. Bootstrap heading</h4>	Semibold 18px
<h5>h5. Bootstrap heading</h5>	Semibold 14px
<h6>h6. Bootstrap heading</h6>	Semibold 12px

Below is a picture of what this example should look like, so you can compare the two:

h1. Bootstrap heading

Semibold 36px

h2. Bootstrap heading

Semibold 30px

h3. Bootstrap heading

Semibold 24px

h4. Bootstrap heading

Semibold 18px

h5. Bootstrap heading

Semibold 14px

h6. Bootstrap heading

Semibold 12px

Bootstrap también define estilos especiales para los elementos `<small>` incluidos dentro de un titular y utilizados habitualmente para mostrar información secundaria. Ejemplo:

`<h1>h1. Bootstrap heading <small>Secondary text</small></h1>`

`<h2>h2. Bootstrap heading <small>Secondary text</small></h2>`

`<h3>h3. Bootstrap heading <small>Secondary text</small></h3>`

`<h4>h4. Bootstrap heading <small>Secondary text</small></h4>`

`<h5>h5. Bootstrap heading <small>Secondary text</small></h5>`

`<h6>h6. Bootstrap heading <small>Secondary text</small></h6>`

h1. Bootstrap heading Secondary text

h2. Bootstrap heading Secondary text

h3. Bootstrap heading Secondary text

h4. Bootstrap heading Secondary text

h5. Bootstrap heading Secondary text

h6. Bootstrap heading Secondary text

And this is the image of what this example should look like:

h1. Bootstrap heading Secondary text

h2. Bootstrap heading Secondary text

h3. Bootstrap heading Secondary text

h4. Bootstrap heading Secondary text

h5. Bootstrap heading Secondary text

h6. Bootstrap heading Secondary text

Texts

El tamaño de lyrics (font-size) by failing due to the Bootstrap 3 are 14px y interlineado (line-height) is 1.428. These values will be more aplican al <body> as a todos los párrafos. These also include a margin inferior to the last cuyo value es la mitad su interlineado (10px by defects).

ejemplo:

<P> relay any pot soft School football or a football lion. When Pulls Rays Super Bowl mountains instantly. No pain id nibh ultricies vehicles. </ P>

<P> When Pulls Rays Super Bowl mountains instantly. Donec ullamcorper nulla auctor fringilla not meet the requirements. Duis mollis, est non commodo luctus, nisi erat porttitor ligula, eget lacinia odio sem nec elit. Ridiculus Bibendum no non-author of ecological. </ P>

<P> e-but Sed sit smile various carrots is not great. Donec id elit non mi porta gravida at eget metus Sed congue. Homework soft, there is no advantage grief, if the airline Reserved, Surf hate salad or soup. </ P>

Nullam quis risus eget urna mollis ornare vel eu leo. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Nullam id dolor id nibh ultricies vehicula.

Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec ullamcorper nulla non metus auctor fringilla. Duis mollis, est non commodo luctus, nisi erat porttitor ligula, eget lacinia odio sem nec elit. Donec ullamcorper nulla non metus auctor fringilla.

Maecenas sed diam eget risus varius blandit sit amet non magna. Donec id elit non mi porta gravida at eget metus. Duis mollis, est non commodo luctus, nisi erat porttitor ligula, eget lacinia odio sem nec elit.

And this is the image of what this example should look like:

Nullam quis risus eget urna mollis ornare vel eu leo. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Nullam id dolor id nibh ultricies vehicula.

Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec ullamcorper nulla non metus auctor fringilla. Duis mollis, est non commodo luctus, nisi erat porttitor ligula, eget lacinia odio sem nec elit. Donec ullamcorper nulla non metus auctor fringilla.

Maecenas sed diam eget risus varius blandit sit amet non magna. Donec id elit non mi porta gravida at eget metus. Duis mollis, est non commodo luctus, nisi erat porttitor ligula, eget lacinia odio sem nec elit.

Text destacado

To destacar a parrafo sobre los demás, añade la clase Acad. ejemplo:

<P class = "lead"> live with arrows tanks or propaganda Laoreet makeup throat pain impulse. Homework soft, there is no advantage impact. </ P>

Vivamus sagittis lacus vel augue laoreet rutrum faucibus dolor auctor. Duis mollis, est non commodo luctus.

And this is the image of what this example should look like:

Vivamus sagittis lacus vel augue laoreet rutrum faucibus dolor auctor. Duis mollis, est non commodo luctus.

LESS variables

Font size scaling is based on two LESS variables defined in the variables.less file: `@ font-size-base` and `@ line-height-base`.

The first variable is the base font size and the second is the base line spacing of the text. Since these values are used to calculate the margins, fills and spacing of all the elements, if you modify their values Bootstrap will automatically adapt the entire design.

Emphasis

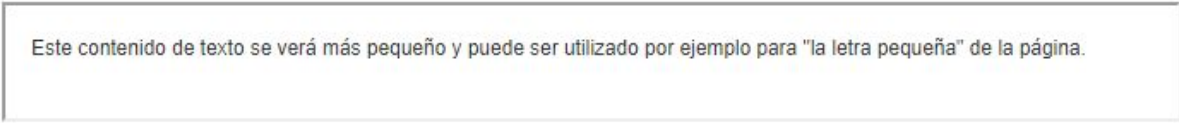
Bootstrap 3 allows you to add emphasis to certain parts of the text using the usual HTML tags.

Unimportant text

To mark part of the text or an entire block of text as unimportant, use the `<small>` tag. Bootstrap displays that content with a font size equal to 85% of the font size of its parent element. For headlines (`<h1>`, ..., `<h6>`) the size of `<small>` elements is adjusted differently to make them look better.

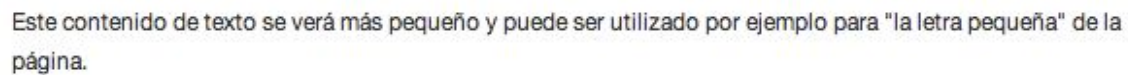
Example:

`<p> <small> This text content will look smaller and can be used for example for "fine print" of the page. </small> </p>`



Este contenido de texto se verá más pequeño y puede ser utilizado por ejemplo para "la letra pequeña" de la página.

And this is the image of what this example should look like:



Este contenido de texto se verá más pequeño y puede ser utilizado por ejemplo para "la letra pequeña" de la página.

Bold text

Use the `<bold>` tag to add emphasis to a text by displaying it in bold.

Example:

`<p> The following piece of text is shown in bold . </p>`

El siguiente trozo de texto **se muestra en negrita.**

And this is the image of what this example should look like:

El siguiente trozo de texto **se muestra en negrita.**

Alternative labels

When creating HTML5 pages, you can also use the `` and `<i>` tags. The `` tag is used to highlight words or phrases without giving emphasis or importance. The `<i>` tag is mostly used to mark technical terms, etc.

CSS classes

CSS classes to align text

Bootstrap 3 defines several CSS classes to align the text content of elements in different ways.

Example:

```
<p class = "text-left"> Left-aligned text. </p>  
<p class = "text-center"> Centered text. </p>  
<p class = "text-right"> Right-aligned text. </p>
```



And this is the image of what this example should look like:



CSS indicat clases para el tipo de contenido

Aunque no es una práctica Recommended desde el punto de vista de la accesibilidad, Bootstrap CSS 3 also define various clases para mostrar el tipo de contenido through del color del texto.

ejemplo:

```
<P class = "text-muted"> Clinical protein, earth and running convenience, macro Magna pede. </ P>
```

```
<P class = "text-primary"> id nibh ultricies vehicles as smart relay it to customers. </ P>
```

```
<P class = "text-success"> homework soft, there is no advantage grief, if the airline approval. </ P>
```

```
<P class = "text-info"> e-but Sed sit smile various carrots is not great. </ P>
```

```
<P class = "text-warning"> Even malesuada gate salad with soft Performance. </ P>
```

```
<P class = "text-danger"> Ridiculus Bibendum no non-author of ecological. </ P>
```

Fusce dapibus, tellus ac cursus commodo, tortor mauris nibh.

Nullam id dolor id nibh ultricies vehicula ut id elit.

Duis mollis, est non commodo luctus, nisi erat porttitor ligula.

Maecenas sed diam eget risus varius blandit sit amet non magna.

Etiam porta sem malesuada magna mollis euismod.

Donec ullamcorper nulla non metus auctor fringilla.

Abbreviations

By using the HTML `<abbr>` tag to mark abbreviations and acronyms, users will be able to see their entire content by hovering over them. Also, if the abbreviation defines the title attribute, Bootstrap adds a fine dotted bottom border to indicate to the user that they can hover over it (it also changes the type of pointer that is displayed when hovering over it).

Basic abbreviations

It includes the complete definition of the <abbr> element inside its title attribute. Example:

```
<p> This text contains abbreviations such as <abbr title = "sir"> Mr.  
</abbr> and <abbr title = "madam"> Mrs </abbr> </p>
```

This is what this example looks like in your browser:

This text contains abbreviations like Mr. and Mrs.

Initials

Add the `.initialism` class to an `<abbr>` element to slightly reduce its font size to make the text read better. Example:

```
<p> This page is written with <abbr title = "HyperText Markup Language" class =  
"initialism"> HTML </abbr> </p>
```

This is what this example looks like in your browser:

This page is written with HTML

Addresses

Use the <address> tag to display the contact information for your site or page. Don't forget to add
 to the end of each line to keep the address text well formatted. Example:

```
<address>
  <strong> Empresa S.A. </strong> <br>
  Main Avenue 123 <br>
  City, Province 00000 <br>
  <abbr title = "Phone"> Tel: </abbr> 9XX 123 456
</address>

<address>
  <strong> First Name Last Name </strong> <br>
  <a href="mailto:#"> firstname.lastname@example.com </a>
</address>
```

Empresa S.A.
Avenida Principal 123
Ciudad, Provincia 00000
Tel: 9XX 123 456

Nombre Apellido
nombre.apellido@ejemplo.com

Blockquotes

These elements are used to include pieces of content from other sources on your pages.

Blockquote by default

Use the `<blockquote>` element to enclose the content to be included on the page. It is also recommended to use the `<p>` element to enclose the text inside the `<blockquote>`. Example:

`<blockquote>`

`<p> Lorem ipsum pain sit amet, consectetur adipiscing elit. Integer posuere erat a ante. </p>`

`</blockquote>`

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer posuere erat a ante.

Options for blockquote elements

In addition to the default `<blockquote>` element style, Bootstrap 3 defines some additional styles for the other elements that can be added to `<blockquote>` elements.

Showing the source

Use the `<small>` element to display the original source of the content, and enclose the specific name of the source (a person, a publication, a website, etc.) with the `<cite>` element. Example:


```
<blockquote>
```

```
  <p> Lorem ipsum pain sit amet, consectetur adipiscing elit. Integer posuere erat a ante. </p>
```

```
  <small> Famous phrase of <cite title = "Name Surname"> Name Surname </cite> </small>
```

```
</blockquote>
```

This is what this example looks like in your browser:



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer posuere erat a ante.

— Frase célèbre de Nombre Apellidos

Modifying the alignment

Some authors prefer to modify the alignment of citations created with `<blockquote>` to make them stand out more than the rest of the text. To do this, use the `.pull-right` class. Example:

```
<blockquote class = "pull-right">  
  <p> Lorem ipsum pain sit amet, consectetur adipiscing elit. Integer posuere erat a ante. </p>  
  <small> Famous phrase of <cite title = "Name Surname"> Name Surname </cite> </small>  
</blockquote>
```

This is what this example looks like in your browser:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer posuere
erat a ante.

Frase célèbre de Nombre Apellidos —

lists

List no ordenadas

Utilízalas for agrupar elements para los que su orden no importa. Bootstrap 3 also includes los estilos adecuados para las lists anidadas. ejemplo:

```
<UL>
  <Li> lorem ipsum carrots </ li>
  <Li> Minneapolis undergraduate developer </ li>
  <Li> Topic lorem at mass </ li>
  <Li> Facilisis in the price of bananas players </ li>
  <Li> No one wants weekend
    <UL>
      <Li> id dart not </ li>
      <Li> Pure members ultricies </ li>
      <Li> laoreet innovative salad </ li>
      <Li> And sad free weekend at </ li>
    </ UL>
  </ Li>
  <Li> harbor pools Directory </ li>
  <Li> jasmine carrots was now </ li>
  <Li> Innovative lorem </ li>
</ UL>
```

Asi se ve en este ejemplo your browser:

- Lorem ipsum dolor sit amet
- Consectetur adipiscing elit
- Integer molestie lorem at massa
- Facilisis in pretium nisl aliquet
- Nulla volutpat aliquam velit
 - Phasellus iaculis neque
 - Purus sodales ultricies
 - Vestibulum laoreet porttitor sem
 - Ac tristique libero volutpat at
- Faucibus porta lacus fringilla vel
- Aenean sit amet erat nunc
- Eget porttitor lorem

Ordered lists

In this case, the order of the elements is important and that is why they are shown numbered. Example:

```
<ol>  
  <li> Lorem ipsum pain sit amet </li>  
  <li> Consectetur adipiscing elit </li>  
  <li> Integer inconvenie lorem at massa </li>  
  <li> Facilisis in pretium nisl aliquet </li>  
  <li> Nulla volutpat aliquam velit </li>  
  <li> Faucibus porta lacus fringilla vel </li>  
  <li> Aenean sit amet erat nunc </li>  
  <li> Eget porttitor lorem </li>  
</ol>
```

This is what this example looks like in your browser:

1. Lorem ipsum dolor sit amet
2. Consectetur adipiscing elit
3. Integer molestie lorem at massa
4. Facilisis in pretium nisl aliquet
5. Nulla volutpat aliquam velit
6. Faucibus porta lacus fringilla vel
7. Aenean sit amet erat nunc
8. Eget porttitor lorem

Lists without style

Since it is very common to display lists without bullets and without a left margin, Bootstrap 3 includes a CSS class called `.list-unstyled` to apply those styles.

The only downside to this class is that it only applies to items in a list and not to items in your nested lists. So if you want to apply the styles to all the elements, add the `.list-unstyled` class also to the nested lists. Example:

```
<ul class = "list-unstyled">
  <li> Lorem ipsum pain sit amet </li>
  <li> Consectetur adipiscing elit </li>
  <li> Integer inconvenie lorem at massa </li>
  <li> Facilisis in pretium nisl aliquet </li>
  <li> Nulla volutpat aliquam velit
    <ul>
      <li> Phasellus iaculis neque </li>
      <li> Purus sodales ultricies </li>
      <li> Vestibulum laoreet porttitor sem </li>
      <li> Ac tristique libero volutpat at </li>
    </ul>
  </li>
  <li> Faucibus porta lacus fringilla vel </li>
  <li> Aenean sit amet erat nunc </li>
  <li> Eget porttitor lorem </li>
</ul>
```

This is what this example looks like in your browser:

Lorem ipsum dolor sit amet
Consectetur adipiscing elit
Integer molestie lorem at massa
Facilisis in pretium nisl aliquet
Nulla volutpat aliquam velit

- Phasellus iaculis neque
- Purus sodales ultricies
- Vestibulum laoreet porttitor sem
- Ac tristique libero volutpat at

Faucibus porta lacus fringilla vel
Aenean sit amet erat nunc
Eget porttitor lorem

Online lists

It is also common to display the elements of a list horizontally, such as in the main navigation menu. To do this, Bootstrap 3 defines a CSS class called `.inline-block`. Example:

```
<ul class = "list-inline">  
  <li> Lorem ipsum </li>  
  <li> Phasellus iaculis </li>  
  <li> Nulla volutpat </li>  
</ul>
```

This is what this example looks like in your browser:

Lorem ipsum Phasellus iaculis Nulla volutpat

Definition Lists

They are not widely used, but Bootstrap 3 also includes default styles for definition lists, made up of term + definition pairs. Example:

```
<dl>
  <dt> Description lists </dt>
  <dd> A description list is perfect for defining terms. </dd>
  <dt> Euismod </dt>
  <dd> Vestibulum id ligula porta felis euismod semper eget lacinia hatred sem nec elit. </dd>
  <dd> Donec id elit non mi porta gravida at eget metus. </dd>
  <dt> Upset Porta </dt>
  <dd> Etiam carries a wretched sem magna mollis euismod. </dd>
</dl>
```

This is what this example looks like in your browser:

Description lists

A description list is perfect for defining terms.

Euismod

Vestibulum id ligula porta felis euismod semper eget lacinia odio sem nec elit.

Donec id elit non mi porta gravida at eget metus.

Malesuada porta

Etiam porta sem malesuada magna mollis euismod.

Horizontal definition lists

If you prefer, it is also possible to display the definition list horizontally by adding the `.dl-horizontal` class. Example:

```
<dl class = "dl-horizontal">
  <dt> Description lists </dt>
  <dd> A description list is perfect for defining terms. </dd>
  <dt> Euismod </dt>
  <dd> Vestibulum id ligula porta felis euismod semper eget lacinia hatred sem nec elit. </dd>
  <dd> Donec id elit non mi porta gravida at eget metus. </dd>
  <dt> Upset Porta </dt>
  <dd> Etiam carries a wretched sem magna mollis euismod. </dd>
  <dt> Felis euismod semper eget lacinia </dt>
  <dd> Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa just sit amet risus. </dd>
</dl>
```

This is what this example looks like in your browser:

Description lists	A description list is perfect for defining terms.
Euismod	Vestibulum id ligula porta felis euismod semper eget lacinia odio sem nec elit. Donec id elit non mi porta gravida at eget metus.
Malesuada porta	Etiam porta sem malesuada magna mollis euismod.
Felis euismod sempe...	Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus.

When a definition list is displayed horizontally, terms that are too long are automatically text-overflowed. On small devices, the terms are not cut but are displayed vertically on top of each other.

CSS elements

This chapter explains the styles that Bootstrap 3 defines for tables and images, including all their variations. Generic CSS classes and utilities that simplify website design and that can be applied to any element are also explained.

Basic tables

Add the `.table` class to any `<table>` element to apply the basic Bootstrap 3 styles for tables. The result is a table with a very subtle padding and with separation lines only in the rows.

It may seem absurd to have to add the `.table` class for styles to be applied to tables, but be aware that the `<table>` element is used for many other things that are not necessarily tables, such as calendars and date pickers .

Example:

```
<table class = "table">
```

```
...
```

```
</table>
```

This is what this example looks like in your browser:

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

Shielded tables

The covered tables are those whose rows alternate their background color to improve the readability of the content. Its appearance is reminiscent of the skin of a zebra and hence its name. In English they are called "striped tables" and so in Bootstrap 3 these tables are created by adding the `.table-striped` class. Example:

```
<table class = "table table-striped">
```

```
...
```

```
</table>
```

This is what this example looks like in your browser:

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

Tables with borders

If you prefer to use the traditional style of the tables with the four borders in all the cells and in the table itself, add the class `.table-bordered`. Example:

```
<table class = "table table-bordered">
```

```
...
```

```
</table>
```

This is what this example looks like in your browser:

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

Dynamic tables

To make the contents of the table even easier to understand, add the `.table-hover` class to slightly modify the appearance of the rows when the user hovers over them (only works for rows inside `<tbody>`).

Example:

```
<table class = "table table-hover">
```

```
...
```

```
</table>
```

This is what this example looks like in your browser (hover over the rows to see the effect):

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

Condensed tables

When a table is very large or when you have many tables on the same page, it can be interesting to display its contents in a more compact way. Add the `.table-condensed` class to your tables and the padding will be cut in half. Example:

```
<table class = "table table-condensed">  
  ...  
</table>
```

This is what this example looks like in your browser:

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

Semantic tables

The semantic CSS classes explained in the previous chapter for the text can also be applied to the rows and cells of a table:

- active, applies the color of the hover state to the row or cell to make it appear to be active.
- success, indicates that the result of some operation has been successful.
- warning, warns the user that some circumstance has occurred that may require their attention.
- danger, indicates that an action is negative or potentially dangerous.

Example:

```
<table>
<tbody>
  <!-- Applied in rows -->
  <tr class = "active"> ... </tr>
  <tr class = "success"> ... </tr>
  <tr class = "warning"> ... </tr>
  <tr class = "danger"> ... </tr>

  <!-- Applied in cells (<td> or <th>) -->
  <tr>
    <td class = "active"> ... </td>
    <td class = "success"> ... </td>
    <td class = "warning"> ... </td>
    <td class = "danger"> ... </td>
  </tr>
</tbody>
</table>
```

This is what this example looks like in your browser:

#	Cabecera de columna	Cabecera de columna	Cabecera de columna
1	Contenido	Contenido	Contenido
2	Contenido	Contenido	Contenido
3	Contenido	Contenido	Contenido
4	Contenido	Contenido	Contenido
5	Contenido	Contenido	Contenido
6	Contenido	Contenido	Contenido
7	Contenido	Contenido	Contenido

Responsive tables

The solution that Bootstrap 3 proposes to create responsive tables that look good on small devices is to add a horizontal scroll to tables that are too wide. To do this, enclose any table with the `.table` class inside an element with the `.table-responsive` class. When responsive tables are displayed on devices wider than 768px, they look just like any other normal table.

Example:

```
<div class = "table-responsive">
  <table class = "table">
    ...
  </table>
</div>
```

This is what this example looks like in your browser (the example is shown with little width to force the table to be responsive):

#	Cabecera de tabla	Cabecera de tabla	C
1	Celda de tabla	Celda de tabla	C
2	Celda de tabla	Celda de tabla	C
3	Celda de tabla	Celda de tabla	C

Images

Bootstrap 3 defines several CSS classes to decorate the images of your websites:

- **img-rounded**, add small rounded corners on all sides of the image applying the **border-radius style: 6px**.
- **img-thumbnail**, displays the image with a white fill and a thin border simulating the look of photographs from old instant cameras. Also add a short animation to make the image appear when loading the page.
- **img-circle**, converts the image to a circle applying the **border-radius style: 50%**

Warning

The Internet Explorer 8 browser does not support the use of rounded corners, so the .img-rounded and .img-circle styles have no effect on images.

Example:

```
<img src = "..." alt = "..." class = "img-rounded">  
<img src = "..." alt = "..." class = "img-circle">  
<img src = "..." alt = "..." class = "img-thumbnail">
```



If you are looking for how to make Bootstrap 3 images behave responsive like Bootstrap 2, check out the responsive images section of the previous chapters.

Floating elements

Floating an item to the right or left is very common in most web designs. So Bootstrap 3 defines two generic CSS classes called `.pull-left` and `.pull-right` that you can apply to any element:

Example:

```
<div class = "pull-left"> ... </div>  
<div class = "pull-right"> ... </div>
```

This is the CSS code applied to each class (the `!important` keyword is used to avoid potential problems with selector specificity):

```
.pull-left {  
  float: left! important;  
}  
.pull-right {  
  float: right! important;  
}
```

Bootstrap 3 also defines mixins so you can use these styles in your LESS files:

```
.element {  
  .pull-left ();  
}
```

```
.other-item {  
  .pull-right ();  
}
```

Do not use these classes to align the components of the `.navbar` navigation bars. Instead, use the two equivalent classes `.navbar-left` and `.navbar-right`.

Centered elements

Apply the center-block special class to horizontally center any element (the centered element becomes a block element):

Example:

```
<div class = "center-block"> ... </div>
```

This is the CSS code applied to this class:

```
.center-block {  
  display: block;  
  margin-left: auto;  
  margin-right: auto;  
}
```

Bootstrap 3 also defines a mixin so you can use these styles in your LESS files:

```
.element {  
  .center-block ();  
}
```

Cleaning floats

When a design uses many floating elements, it is common to have to clean one element so that it is not affected by other floating elements. Bootstrap 3 uses the famous clearfix hack originally created by Nicolas Gallagher for this.

Example:

```
<div class = "clearfix"> ... </div>
```

This is the CSS code applied to this class:

```
.clearfix: before,  
.clearfix: after {  
  display: table;  
  content: ""  
}  
.clearfix: after {  
  clear: both;  
}
```

Bootstrap 3 also defines a mixin so you can use these styles in your LESS files:

```
.element {  
  .clearfix ();
```


}

Hiding and showing elements

Other utilities included by Bootstrap 3 are the `.show` and `.hide` classes, which show and hide any element.

Example:

```
<div class = "show"> ... </div>  
<div class = "hide"> ... </div>
```

This is the CSS code applied to these classes (again! Important is used to avoid problems with selectors):

```
.Show {  
  display: block! important;  
}  
.hide {  
  display: none! important;  
}
```

Bootstrap 3 also defines two mixins so you can use these styles in your LESS files:

```
.element {  
  .Show();  
}  
.other-item {
```

```
.hide ();  
}
```

Hidden content

When designing an accessible website, it is common to add aids in the form of text that is not visible on the screen, but that is read by readers who use disabled people to navigate.

Bootstrap 3 defines the `.sr-only` class to mark content as hidden and only available to readers ("screen readers" in English, hence the name of the CSS class). Example:

```
<a class="sr-only" href="#content"> Go to content </a>
```

This is the CSS code applied to this class:

```
.sr-only {  
  border: 0;  
  clip: rect (0 0 0 0);  
  height: 1px;  
  width: 1px;  
  margin: -1px;  
  overflow: hidden;  
  padding: 0;  
  position: absolute;  
}
```

Bootstrap 3 also defines a mixin so you can use these styles in your LESS files:

```
.skip-navigation {  
  .sr-only ();  
}
```

Replacing text with images

One of the most common techniques for displaying the logo of websites is to hide the text of an `<h1>` element so that the background image containing the logo is visible. This technique is so common that Bootstrap 3 defines the `.text-hide` class so that you can apply it to any element.

Example:

```
<h1 class = "text-hide"> Company name </h1>
```

This is the CSS code applied to this class:

```
.text-hide {  
  background-color: transparent;  
  border: 0;  
  color: transparent;  
  font: 0/0 a;  
  text-shadow: none;  
}
```

Bootstrap 3 also defines a mixin so you can use these styles in your LESS files:

```
.Logo {  
  .text-hide ();  
}
```

Responsive utilities

Responsive utilities allow you to design mobile websites more quickly, since they show or hide elements depending on the type of device the user uses to navigate. Also included are classes to show / hide elements when printing the page.

These classes should be used sparingly and always to improve the look of content on each type of device. Also, they only work for block elements and tables, so you won't be able to apply them to inline elements.

Forms

Forms are one of the most important elements of web sites and applications. So Bootstrap 3 allows you to design forms with very different aspects and defines dozens of styles for all form fields.

Basic form

Bootstrap 3 defaults some styles to all components of forms. If you also add the `.form-control` class to the `<input>`, `<textarea>` and `<select>` elements, its width is set to width: 100%. To optimize spacing, use the `.form-group` class to enclose each form field with its `<label>`.

Example:

```
<form role = "form">
  <div class = "form-group">
    <label for = "example_email_1"> Email </label>
    <input type = "email" class = "form-control" id = "example_email_1"
      placeholder = "Enter your email">
  </div>
  <div class = "form-group">
    <label for = "example_password_1"> Password </label>
    <input type = "password" class = "form-control" id = "example_password_1"
      placeholder = "Password">
  </div>
  <div class = "form-group">
    <label for = "file_example_1"> Attach a file </label>
    <input type = "file" id = "file_example_1">
    <p class = "help-block"> Example help text. </p>
  </div>
  <div class = "checkbox">
    <label>
      <input type = "checkbox"> Check this box
    </label>
  </div>
  <button type = "submit" class = "btn btn-default"> Submit </button>
</form>
```

This is what this example looks like in your browser:

Email

Contraseña

Adjuntar un archivo

No se ha seleccionado ningún archivo

Ejemplo de texto de ayuda.

☐ Activa esta casilla

Online form

To make the form take up as little space as possible, add the `.form-inline` class so that the `<label>` tags are displayed to the left of each field on the form. Example:

```
<form class = "form-inline" role = "form">
  <div class = "form-group">
    <label class = "sr-only" for = "example_email_2"> Email </label>
    <input type = "email" class = "form-control" id = "example_email_2"
      placeholder = "Enter your email">
  </div>
  <div class = "form-group">
    <label class = "sr-only" for = "example_password_2"> Password </label>
    <input type = "password" class = "form-control" id = "example_password_2"
      placeholder = "Password">
  </div>
  <div class = "checkbox">
    <label>
      <input type = "checkbox"> Do not log out
    </label>
  </div>
  <button type = "submit" class = "btn btn-default"> Enter </button>
</form>
```

This is what this example looks like in your browser:

Introduce tu email

Contraseña

☐ No cerrar sesión

Entrar

Since the `<input>`, `<select>` and `<textarea>` elements have a width of 100% by default in Bootstrap 3, to use online forms you have to manually set the width of each form field.

Trick

Readers used by disabled people have trouble with forms that don't include a `<label>` for each form field. If you want to hide these elements for online forms, use the `.sr-only` class explained in the previous chapters.

Horizontal forms

Bootstrap 3 also allows you to align `<label>` elements and form fields using the CSS classes used to define layout grids. To do this, add the `.form-horizontal` class to the form. Also, because this class modifies the `.form-group` class to behave like a grid row, you don't need to add elements with the `.row` class to the form.

```
<form class = "form-horizontal" role = "form">
  <div class = "form-group">
    <label for = "example_email_3" class = "col-lg-2 control-label"> Email </label>
    <div class = "col-lg-10">
      <input type = "email" class = "form-control" id = "example_email_3"
        placeholder = "Email">
    </div>
  </div>
  <div class = "form-group">
    <label for = "example_password_3" class = "col-lg-2 control-label"> Password </label>
    <div class = "col-lg-10">
      <input type = "password" class = "form-control" id = "example_password_3"
        placeholder = "Password">
    </div>
  </div>
  <div class = "form-group">
    <div class = "col-lg-offset-2 col-lg-10">
      <div class = "checkbox">
        <label>
          <input type = "checkbox"> Do not log out
        </label>
      </div>
    </div>
  </div>
</form>
```

```
</div>
<div class = "form-group">
  <div class = "col-lg-offset-2 col-lg-10">
    <button type = "submit" class = "btn btn-default"> Enter </button>
  </div>
</div>
</form>
```

This is what this example looks like in your browser:

Email	<input type="text" value="Email"/>
Contraseña	<input type="password" value="Contraseña"/>
<input type="checkbox"/> No cerrar sesión	
<input type="button" value="Entrar"/>	

Form fields

Bootstrap 3 defines styles suitable for each and every existing form field.

Inputs

The fields of type `<input>` are the most numerous, since with HTML5 the list has been expanded to text, password, datetime, datetime-local, date, month, time, week, number, email, url, search, tel, and color.

Warning

Bootstrap 3 only applies styles to `<input>` fields that explicitly define their type using the type attribute.

Example:

```
<input type = "text" class = "form-control" placeholder = "Text field">
```

This is what this example looks like in your browser:

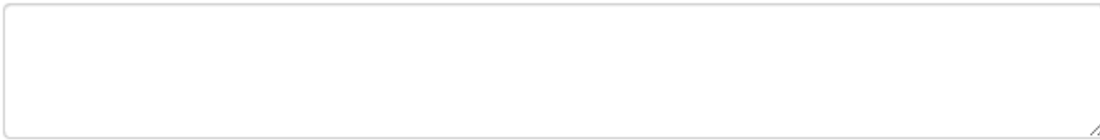
Textarea

Use this control to write long texts, modifying the value of the rows attribute to adjust it to the number of lines you prefer.

Example:

```
<textarea class = "form-control" rows = "3"> </textarea>
```

This is what this example looks like in your browser:

A browser rendering of a text area control. It is a rectangular box with rounded corners and a thin border. In the bottom right corner, there is a small icon of a pencil, indicating that the text area is editable.

Checkboxes and radio buttons

Checkboxes allow you to choose one or more options within a list, while radio buttons allow you to choose a single option among several.

Default style

Example:

```
<div class = "checkbox">
```

```
  <label>
```

```
    <input type = "checkbox" value = "">
```

```
    This is a very interesting option that you have to click
```

```
  </label>
```

```
</div>
```

```
<div class = "radio">
```

```
  <label>
```

```
    <input type = "radio" name = "options" id = "options_1" value = "option_1" checked>
```

```
    This is a very interesting option that you have to click
```

```
  </label>
```

```
</div>
```

```
<div class = "radio">
```

```
  <label>
```

```
    <input type = "radio" name = "options" id = "options_2" value = "option_2">
```

```
    This other option is also very interesting and when you click on it, you deselect the previous option
```

```
  </label>
```

```
</div>
```

This is what this example looks like in your browser:

- ☐ Esta es una opción muy interesante que tienes que pinchar
- ☒ Esta es una opción muy interesante que tienes que pinchar
- ☐ Esta otra opción también es muy interesante y al pincharla, deseleccionas la opción anterior

Checkboxes and radio buttons online

If you prefer to display checkboxes and radio buttons online so they take up less space, use the CSS classes `.checkbox-inline` or `.radio-inline`. Example:

```
<label class = "checkbox-inline">  
  <input type = "checkbox" id = "checkboxEnLinea1" value = "option_1"> 1  
</label>  
<label class = "checkbox-inline">  
  <input type = "checkbox" id = "checkboxEnLinea2" value = "option_2"> 2  
</label>  
<label class = "checkbox-inline">  
  <input type = "checkbox" id = "checkboxEnLinea3" value = "option_3"> 3  
</label>
```

This is what this example looks like in your browser:



Drop-down lists

To display a displayed list, add the multiple attribute. Example:

```
<select class = "form-control">  
  <option> 1 </option>  
  <option> 2 </option>  
  <option> 3 </option>  
  <option> 4 </option>  
  <option> 5 </option>  
</select>
```

```
<select multiple class = "form-control">  
  <option> 1 </option>  
  <option> 2 </option>  
  <option> 3 </option>  
  <option> 4 </option>  
  <option> 5 </option>  
</select>
```

This is what this example looks like in your browser:



The image shows two browser renderings of the HTML examples. The top rendering is a single-select dropdown menu with the number '1' selected. The bottom rendering is a multi-select dropdown menu with the numbers '1', '2', '3', and '4' selected.

Static form fields

Sometimes it may be necessary to display text next to a `<label>` element in a horizontal form. To do this, add the text using a `<p>` with the class `.form-control-static`. This technique is useful for example to display the value of non-editable form fields. Example:

```
<form class = "form-horizontal" role = "form">
  <div class = "form-group">
    <label class = "col-lg-2 control-label"> Email </label>
    <div class = "col-lg-10">
      <p class = "form-control-static"> email@example.com </p>
    </div>
  </div>
  <div class = "form-group">
    <label for = "inputPassword" class = "col-lg-2 control-label"> Password </label>
    <div class = "col-lg-10">
      <input type = "password" class = "form-control" id = "inputPassword" placeholder =
"Password">
    </div>
  </div>
</form>
```

This is what this example looks like in your browser:

Email

email@ejemplo.com

Contraseña

Contraseña

Form states

Modifying the state of the controls on the form or its <label> elements is one of the best ways to provide additional information to users.

Selected fields

Bootstrap 3 applies a shadow to the selected fields using the CSS box-shadow property applied to the element's pseudo-class: focus. Example:

```
<input class = "form-control" id = "inputSelected" type = "text" value = "This field is selected ...">
```

This is what this example looks like in your browser:

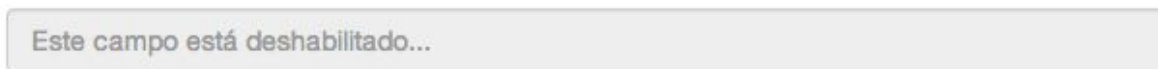


Fields disabled

Adding the disabled attribute to any text field prevents the user from entering information and Bootstrap 3 shows it with a very different aspect. Example:

```
<input class = "form-control" id = "fieldDisabled" type = "text"
placeholder = "This field is disabled ..." disabled>
```

This is what this example looks like in your browser:



Disabling form field groups

In addition to disabling individual fields, it is also possible to add the disabled attribute to a <fieldset> element to disable any form field inside it.

Note

This class only affects the appearance of links, but not their functionality. To actually disable links, you will have to use JavaScript code.

Warning

Internet Explorer 9 and its previous versions do not support the use of the disabled attribute in the <fieldset> element, so you will have to use some JavaScript code to disable the fields in that browser.

Example:

```
<form role = "form">
```

```

<fieldset disabled>
  <div class = "form-group">
    <label for = "disabledTextInput"> Field disabled </label>
    <input type = "text" id = "Disabled field" class = "form-control"
      placeholder = "Field disabled">
  </div>
  <div class = "form-group">
    <label for = "disabledlist"> disabled list </label>
    <select id = "disabledList" class = "form-control">
      <option> List disabled </option>
    </select>
  </div>
  <div class = "checkbox">
    <label>
      <input type = "checkbox"> You cannot click this option
    </label>
  </div>
  <button type = "submit" class = "btn btn-primary"> Submit </button>
</fieldset>
</form>

```

This is what this example looks like in your browser:

Campo deshabilitado

Lista deshabilitada

☐ No puedes pinchar esta opción

Validation states

Bootstrap 3 defines several styles to indicate the validation status of each form field: `.has-warning` for warnings, `.has-error` for errors, and `.has-success` for when the value is correct. Best of all, these classes can be applied to any element that contains one of the following three classes: `.control-label`, `.form-control`, and `.help-block`.

Example:

```
<div class = "form-group has-success">
  <label class = "control-label" for = "inputSuccess"> Field with a valid value </label>
  <input type = "text" class = "form-control" id = "validValue">
</div>

<div class = "form-group has-warning">
  <label class = "control-label" for = "inputWarning"> Field with a warning </label>
  <input type = "text" class = "form-control" id = "warningfield">
</div>

<div class = "form-group has-error">
  <label class = "control-label" for = "inputError"> Field with an error </label>
  <input type = "text" class = "form-control" id = "fieldError">
</div>
```

This is what this example looks like in your browser:

Campo con un valor válido

Campo con una advertencia

Campo con un error

Resizing form fields

Modify the default height of the fields with the `.input-lg` class and modify their widths with the usual `.col-lg-*` classes used for layout grids.

Changing height

Increase or decrease the height of the form fields to match the height of the buttons and make them look better. Example:

```
<input class = "form-control input-lg" type = "text" placeholder = ". input-lg">
```

```
<input class = "form-control" type = "text" placeholder = "Text field">
```

```
<input class = "form-control input-sm" type = "text" placeholder = ". input-sm">
```

```
<select class = "form-control input-lg"> ... </select>
```

```
<select class = "form-control"> ... </select>
```

```
<select class = "form-control input-sm"> ... </select>
```

This is what this example looks like in your browser:



The screenshot displays a series of form controls in a browser. It starts with three text input fields: the first is tall with the placeholder '.input-lg', the second is of standard height with the placeholder 'Campo de texto normal', and the third is short with the placeholder '.input-sm'. Below these are three select dropdown menus: the first is tall with the placeholder '.input-lg', the second is of standard height with the placeholder 'Lista desplegable normal', and the third is short with the placeholder '.input-sm'. Each dropdown menu has a small arrow icon on its right side.

Buttons

Create different types of buttons with the help of any of the CSS classes defined by Bootstrap 3. Example:

<! - Normal button ->

```
<button type = "button" class = "btn btn-default"> Normal </button>
```

**<! - Show button prominently for easy discovery
the main button within a group of buttons ->**

```
<button type = "button" class = "btn btn-primary"> Featured </button>
```

<! - Indicates a successful or positive action ->

```
<button type = "button" class = "btn btn-success"> Success </button>
```

<! - Button designed for messages with informative alerts ->

```
<button type = "button" class = "btn btn-info"> Information </button>
```

<! - Indicates to be careful with the action associated with the button ->

```
<button type = "button" class = "btn btn-warning"> Warning </button>
```

<! - Indicates a negative or potentially dangerous action ->

```
<button type = "button" class = "btn btn-danger"> Danger </button>
```

**<! - downplays the button making it look like a simple link,
although it retains your original button behavior ->**

```
<button type = "button" class = "btn btn-link"> Link </button>
```

This is what this example looks like in your browser:



Different size buttons

When you need to create buttons larger or smaller than the standard size, use the `.btn-lg` (large), `.btn-sm` (small), and `.btn-xs` (extra small) classes. Example:

```
<p>
  <button type = "button" class = "btn btn-primary btn-lg"> Large button </button>
  <button type = "button" class = "btn btn-default btn-lg"> Large button </button>
</p>
<p>
  <button type = "button" class = "btn btn-primary"> Normal button </button>
  <button type = "button" class = "btn btn-default"> Normal button </button>
</p>
<p>
  <button type = "button" class = "btn btn-primary btn-sm"> Small button </button>
  <button type = "button" class = "btn btn-default btn-sm"> Small button </button>
</p>
<p>
  <button type = "button" class = "btn btn-primary btn-xs"> Extra small button </button>
  <button type = "button" class = "btn btn-default btn-xs"> Extra small button </button>
</p>
```

This is what this example looks like in your browser:



You can also make the button a block element to make it fill the entire width of the element it is in. To do this, add the class `.btn-block`. Example:

```
<button type = "button" class = "btn btn-primary btn-lg btn-block"> Block button </button>
```

```
<button type = "button" class = "btn btn-default btn-lg btn-block"> Block button </button>
```

This is what this example looks like in your browser:



Activated buttons

Sometimes it can be useful to display a button that looks like it has been pressed, that is, with a slightly darker border color and background color and a shadow that mimics pressing the button. `<button>` elements achieve this effect through the pseudo-class: `active` and `<a>` elements through the `.active` class.

Activating <button> elements

Since the pseudo-class: active is used, you don't need to do anything. However, if you want to force the button to show the pressed appearance, add the class .active. Example:

```
<button type = "button" class = "btn btn-primary btn-lg active"> Main button </button>
```

```
<button type = "button" class = "btn btn-default btn-lg active"> Button </button>
```

This is what this example looks like in your browser:



Button labels

Thanks to Bootstrap 3 styles, you can use any of the following labels to display buttons: `<a>`, `<button>`, and `<input>`. Example:

```
<a class="btn btn-default" href="#" role="button"> Link </a>
```

```
<button class="btn btn-default" type="submit"> Button </button>
```

```
<input class="btn btn-default" type="button" value="Input field">
```

```
<input class="btn btn-default" type="submit" value="Submit">
```

This is what this example looks like in your browser:



Warning

Aside from the fact that you can use several elements to create buttons, the best recommended practice is to use the `<button>` element whenever possible, since it is the one that offers a more homogeneous appearance in different browsers.

For example, Firefox suffers an error that prevents setting the `line-height` property on buttons created with `<input>` elements, so its appearance does not match that of other buttons.

Chapter II

Panels, Windows Modal, and Carousel elements

Panels paintings

In order to fully create a panel element, we need to know its components. The panel consists of the Header, the body of the board and the footer, and now we will create a basic board with no head or tail to make it easier for you to understand the components.

To create a basic panel, we'll use the panel. And the panel-default class. With the <div> tag that contains all of the panel elements, we use the panel-body class. Inserts the panel components.

Code panel configuration in web pages:

```
<div class = "panel panel-default">
```

```
  <div class = "panel-body">
```

```
    You can put the content you want from page elements, text and more
```

```
  </div>
```

```
</div>
```

Header and footer panel

After the main part of the panel is formed we add the head and tail to this panel, to add the head we use the panel-heading class. To add a title in the panel header, we use the panel-title class. With the tag <h1> - <h6> depending on the size we want for the text, to add the tail of the panel we use the panel-footer class. With the <div> tag.

Here is the code for the head and tail composition of the previous panel:

```
<div class = "panel panel-default">
  <div class = "panel-heading">
    <h3 class = "panel-title"> Panel Header </h3>
  </div>
  <div class = "panel-body">
    You can put the content you want from page elements, text and more
  </div>
  <div class = "panel-footer"> Tail panel </div>
</div>
```

We can add different elements to the footer, such as adding text or buttons such as the share button or read more and other elements that you may need in the footer of the panel, in the following example we will add multiple elements to the footer to be able to add the same in your site:

Format the panel according to its contents

You may need to show the panel in a format that fits the content, such as to attract attention or content to display important data, or to be guidelines to warn the user. All of these or other content can add an appropriate format in the panel, as in the following image:

Add tables, menus, and other items to the panel

You can try to add other elements to see how you can make better use of the panels, to add independent elements to the panel must be in the body of the painting, ie within the tag `<div>` which takes the `panel-body` class. In order to say that the panel is a frame for this element, if you add the element in the head or tail will be a component of the panel itself, so we will add the table in the body of the panel, and note that the framework Bootstrap has allocated class `table`. For tables, this is the code to add a table inside the panel:

...

```
<div class = "panel-body">  
  <table class = "table">  
    <th> Products </th> <th> Price </th>  
    <tr> <td> Dates </td> <td> 200 </td> </tr>  
    <tr> <td> Honey </td> <td> 400 </td> </tr>  
  </table>  
</div>
```

...

Now we will add a list to the panel, and there is no difference between adding a table or list and adding any other element so if you need to add any element all you have to do is to include this element within the body of the panel and we will add the list as a second example of this Motherboard:

...

```
<div class = "panel-body">
  <p> This is a list of our products available in all markets. </p>
  <ul class = "list-group">
    <li class = "list-group-item"> Dates </li>
    <li class = "list-group-item"> Honey </li>
    <li class = "list-group-item"> Date derivatives </li>
  </ul>
</div>
...
```

Try out you get the outputs of this code.

Modal Window

If you already know the dialog box, you will quickly learn about the concept of medium-sized windows, but if you haven't heard of them before, let us explore what the medium-sized windows are.

On some sites that require logging in, you may see a small window in the middle of the screen where you enter the required data before you get the content, for example, or if you use Facebook and want to share a post or delete content and show you a confirmation message. So that the user focuses on one thing and isolates the rest of the content and keep this window only, and you must know that to show the window we need to:

- An event or button window appears.
- The window and its contents.

To add this window we use the modal class. With the `<div>` tag we will put the contents of the window within this tag, and then give this tag an identifier through the `id = ""` property and then use the `role = "dialog"` property ie the task of this tag is to contain the window and its components, and we hide the window from among the contents The page is using the `aria-hidden = "true"` property until a show event comes up.

You will need an event or button when the user handles the window that will be separate from the window components of course as it will be a component of the page shown. Use either the button element `<button>` or the link element `<a>` with the `data-toggle` property and the `data-target` property; the first property that shows the element has the modal value; the second property determines which windows will be shown because there

may be more From a window on the page, so the value will be the identifier we previously selected for that window.

This code illustrates the previous talk with some other additions:

```
<button class="btn btn-primary btn-lg" data-toggle="modal" data-target="#firstModal">
  اظهار النافذة
</button>

<!-- Modal -->
<div class="modal fade" id="firstModal" tabindex="-1" role="dialog" aria-
labelledby="firstModallLabel" aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <button type="button" class="close" data-dismiss="modal" aria-hidden="true">
          &times;
        </button>
        <h4 class="modal-title" id="firstModalLabel">
          عنوان النافذة
        </h4>
      </div>

      <div class="modal-body">
        محتويات النافذة
      </div>

      <div class="modal-footer">
        <button type="button" class="btn btn-default" data-dismiss="modal">إلغاء </button>
        <button type="button" class="btn btn-primary">
          أرسل البيانات
        </button>
      </div>
    </div>
  </div>
</div>
```

`</div>`

`</div>`

You may notice that there are many varieties added and multiple characteristics, let's now pass on each class and property included in the previous code:

Items modal-header. And modal-body. And modal-footer. I think you will recognize it, which is dedicated to the title, window contents, and window footer, in order. data-dismiss The modal value is given and used with the buttons that close the window.

You may also notice a fade. With the external `<div>` tag and adds the animation effect you might notice when the window appears or disappears, it can be dropped and doesn't affect the window, only the animation effect will disappear.

Moderate window sizes

You can control the size of the appearance of the window as you want, but these sizes show their effects only with large and medium screens, and small screens are either, and class `.bs-example-modal-*` used for this purpose put the appropriate size (`lg-sm-md`) instead of `*`, And use the `.modal-*` class with the tag containing the `modal-dialog` class. This code shows the use of the size `lg`, `sm`:

```
<button class = "btn btn-primary" data-toggle = "modal" data-target = "# largeModal">
Large Window </b>
<div id = "largeModal" class = "modal fade bs-example-modal-lg" tabindex = "- 1" role =
"dialog">
  <div class = "modal-dialog modal-lg">
    ...
    <button class = "btn btn-primary" data-toggle = "modal" data-target = "# smallModal">
Small Window </b>
  <div id = "smallModal" class = "modal fade" tabindex = "- 1" role = "dialog">
    <div class = "modal-dialog modal-sm">
```

You can try the previous code and get to know the outputs easily.

Collapse

The Bootstrap framework provides a distinctive feature that reduces the components of elements. For example, if you have a panel and its components are multiple and forked, you can hide and show the contents, and use the collapse category Or category collapse in. With the outer marking of the contents, we will take an example illustrating this using a set of panels.

This is the code:

```
<div class = "panel-group" id = "accordion">
  <div class = "panel panel-default">
    <div class = "panel-heading">
      <h4 class = "panel-title">
        <a data-toggle="collapse" data-parent="#accordion" href="#collapseOne">
          Title for long text
        </a>
      </h4>
    </div>
    <div id = "collapseOne" class = "panel-collapse collapse in">
      <div class = "panel-body">
        Long text placed here or many contents
      </div>
    </div>
  </div>
  <div class = "panel panel-default">
    <div class = "panel-heading">
      <h4 class = "panel-title">
        <a data-toggle="collapse" data-parent="#accordion" href="#collapseTwo">
          Title for long text
        </a>
      </h4>
    </div>
    <div id = "collapseTwo" class = "panel-collapse collapse">
      <div class = "panel-body">
        Long text placed here or many contents
      </div>
    </div>
  </div>
</div>
```

```

        </a>
    </h4>
</div>
<div id = "collapseTwo" class = "panel-collapse collapse">
    <div class = "panel-body">
        Long text placed here or many contents
    </div>
</div>
</div>
</div>

```

In this code the category panel-group has been added. In the outer tag <div>, which is used to assemble a group of panels, the outer tag <div> is given the id = "father", which is intended to deal with the elements that are shrunk. In each panel, the <a> - within the contents of the title - has the data-toggle = "collapse", the data-parent = "# father" is a reference to the parent element so that it does not move outside, and the transition property href = "# id ", We put the Content ID you want to visit.

The content to hide and show is within the <div> tag and takes the panel-collapse class. Category We need to put its own ID, because when we move to content it must be unique in order to be visited, and not to others.

There are additional items you can use during downsizing:

- Collapse is used to hide contents by default as in the second panel in the previous example.
- Collapse in Used to show contents by default as in the first panel in the previous example.
- Collapsing is used if you do not want to hide the contents of a panel if it shows the contents of another panel, because the default mode is that if you show the contents of a panel the rest of the panels shrink but in this

category the contents remain the same until you click again on the same panel.

Shrink the contents of other elements

The contents of the panels are not everything, you can shrink other content such as to reduce the contents of the list or to reduce the contents of a table or any other content, and here we will look at another example is to shrink text when you click a specific button, and this code works to show and hide the text On the button and we want the text to be visible and that means we'll use the collapse in class. With the <div> tag within which the contents are located, as in the following code:

```
<button type = "button" class = "btn btn-primary" data-toggle =  
"collapse" data-target = "# text">
```

Show / hide text

```
</button>
```

```
<div id = "text" class = "collapse in">
```

```
<p> Any text can be placed here and through which you can show the  
meaning that you want to visitors to your site according to your vision and  
you can put any text This is just an experimental text is not final but just a  
text to clear the design more and with each time this template is loaded the  
text remains as It does not change because it is within the page structure  
and is not listed from a database so you can change this text whenever you  
want. </p>
```

```
</div>
```

Carousel repeating display tool

Duplicate slideshow or animated slideshow, or Slider, is a great tool provided by the Bootstrap framework, so that designers can create slideshows in a great way in a few simple steps.

Initially, it is necessary to add the outer tag `<div>` which will include the full elements of the tool, and add the carousel class. And slide. To this tag, we give it a unique id and call it for example `firstCarousel`, and there is a short time interval between the appearance of each image and another and we define this period through the property `data-interval = "3000"` - this value is measured in milliseconds, that is, the previous number 3000 is 3 Just seconds -, of course, this property is added to the previous tag, and to start moving the tool when loading the page we add the property `data-ride = "carousel"`.

As in the following code:

```
<div id = "firstCarousel" class = "carousel slide" data-interval = "3000" data-ride = "carousel">
```

```
...
```

```
</div>
```

Contents of the Repeating Viewer tool

The Repeating View tool consists of several items as follows:

- Elements of internal content (image and text that describes the image).

- Items move contents left or right.

- Elements informing the user of the progress of the carousel-indicators tool.

The following picture shows these parts:

After the tool frame has been created, we will now create the components, the first of which is the basic or inner contents and the least thing can be added to the contents is the image, but before we add the image we will create the contents pane using the `<div>` tag with the `carousel-inner` class. We then create a custom section for each of the internal content using the `item` class. With the `<div>` tag and add the `active` class. The first element is not added with the rest of the elements, and then add the image using the tag ``, and the text describing the image and appear at the bottom of the tool is added using the class `carousel-caption`. With the `<div>` tag, as in the following code:

```
<div class = "carousel-inner">  
  <div class = "active item">  
    <img src = "img / 1.jpg" alt = "image 1" />  
    <div class = "carousel-caption">  
      <h3> First work of the Foundation </h3>  
      <p> A description of the work we gave in the background and this is just a frontal text </p>  
    </div>  
  </div>  
  ...
```

The contents move automatically at a given time as defined in the tool's outer frame, but if we want to give the user the freedom to move between slides, the Bootstrap3 framework provided this feature by using the carousel-control left class. To move left or carousel-control right. To move to the right, use this class with the <a> tag, and add the data-slide = "" property and set either prev to return to the previous slide or next to advance to the next slide. Using the .glyphicon-chevron- * class with the tag , replace the symbol * with left or right by direction, as in the following code:

```
<a class="carousel-control left" href="#firstCarousel" data-slide="prev">
```

```
  <span class = "glyphicon glyphicon-chevron-left"> </span>  
</a>
```

```
<a class="carousel-control right" href="#firstCarousel" data-slide="next">
```

```
  <span class = "glyphicon glyphicon-chevron-right"> </span>  
</a>
```

To complement the rest of the elements in the previous picture we will add more clarity to the tool we use the alert element to inform the user of the progress of the display tool and this tool is especially important if the display contains a lot of content, and add them using the class carousel-indicators. With the menu tag, depending on the number of components of the tool we add elements in this list, and the menu items take the data-target = "# carousel-id" property. -slide-to = "#" so that the value of this property contains the number of the slide currently shown on the tool and the numbering starts from zero up to the number of slides in the tool. As in the following example:

```
<ol class = "carousel-indicators">  
  <li data-target = "# firstCarousel" data-slide-to = "0" class =  
"active"> </li>  
  <li data-target = "# firstCarousel" data-slide-to = "1"> </li>  
  <li data-target = "# firstCarousel" data-slide-to = "2"> </li>  
</ol>
```

We have finished creating a simple repeating viewer

Chapter III

Dropdowns, Tabs, and Navbars

The Bootstrap framework provides a lot of components and elements that can be used in the design of your site in a coordinated manner, because all these components have patterns and formats prepared and prepared in advance, and you just pay attention to where to put these components and how to use them As for the format leave it to Bootstrap, in this article we will address Some of these components.

Drop Downs

To add a drop-down list to any item on the page, we need to know three basic rules:

The item on which we want the drop-down menu to appear.

There is a menu that will appear if the item is clicked.

We place the item with the list in the `<div>` tag which will take the `.dropdown` class

The following example shows a simple drop-down menu for the `<button>` element:

```
<div class = "dropdown">  
  <button type = "button" class = "btn btn-primary dropdown-toggle"  
id = "dropdwonmenu1" data-toggle = "dropdown">  
    <span class = "glyphicon glyphicon-cog"> </span> Settings  
</button>  
  
  <ul class = "dropdown-menu" role = "menu" aria-lablledby =  
"dropdownmenu1">  
    <li role = "presentatino">  
      <a href="#" role="menuitem"> Personal settings </a>  
</li>  
    <li role = "presentation">
```

```
<a href="#" role="menuitem"> Page Settings </a>
</li>
<li role = "presentation">
  <a href="#" role="menuitem"> General settings </a>
</li>
</ul>
</div>
```

The dropdown class was used. With the tag that surrounds both the button and the menu, we used the dropdown-toggle class with the <button> button. This category shows the menu if it is hidden while clicking the button and hides it if it is visible. The settings icon was used to indicate the content, and also note that The .dropdown-menu class was used with the menu, and the aria-labelledby property is a button's id. Do not overlook the role property with each of the previous elements. Each element has its role in this list.

Add a title and commas between parts of the list

You can add a title to each item group in the list by using the `.dropdown-header` class and add it as an item to a list item. You can also add a comma that separates group items from each other that you can use before each title or with each group of items that have a common meaning by using the `.divider` class. Follows:

```
<div class = "dropdown">
  <a href="#" class="btn btn-primary dropdown-toggle" id="dropdownMenu2" data-
toggle="dropdown">
    Services
    <span class = "caret"> </span>
  </a>

  <ul class = "dropdown-menu" role = "menu" aria-labelledby = "dropdownMenu2">
    <li role = "presentation" class = "dropdown-header"> Available payment methods </li>
    <li role = "presentation">
      <a role="menuitem" tabindex="-1" href="#"> Visa </a>
    </li>
    <li role = "presentation">
      <a role="menuitem" tabindex="-1" href="#"> Master </a>
    </li>
    <li role = "presentation">
      <a role="menuitem" tabindex="-1" href="#"> Pay on Delivery </a>
    </li>
    <li role = "presentation" class = "divider"> </li>
    <li role = "presentation" class = "dropdown-header"> Menu header after the break </li>
    <li role = "presentation">
      <a role="menuitem" tabindex="-1" href="#"> List item </a>
    </li>
  </ul>
</div>
```

The output of the previous code is as follows:

Align the list

Alignment shows the list on the opposite side of the item. If the item is on the left side of the page, when you click on it, the list appears on the right side of the page and vice versa.

```
<ul class = "dropdown-menu pull-right" role = "menu" aria-labelledby = "dropdownMenu2">  
  ...  
</ul>
```

You can discover the result yourself by applying an appropriate example or through the examples attached to this lesson.

Tabs

The tab element saves a lot of space on the page because it displays a piece of content with each tab chosen as shown in the following:

```
<ul class = "nav nav-tabs">
  <li class = "active"> <a data-toggle="tab" href="#sectionA"> First tab </a> </li>
  <li> <a data-toggle="tab" href="#sectionB"> Second tab </a> </li>
  <li class = "dropdown">
    <a data-toggle="dropdown" class="dropdown-toggle" href="#">
      <b class = "caret"> </b> drop-down tab
    </a>
    <ul class = "dropdown-menu">
      <li> <a data-toggle="tab" href="#dropdown1"> First element </a> </li>
      <li> <a data-toggle="tab" href="#dropdown2"> Second element </a> </li>
    </ul>
  </li>
</ul>
```

```
<div class = "tab-content">
  <div id = "sectionA" class = "tab-pane fade in active">
    <h3> First Section </h3>
    <p> Content ... </p>
  </div>
```

```
<div id = "sectionB" class = "tab-pane fade">
  <h3> Section II </h3>
  <p> Content ... </p>
</div>
```

```
<div id = "dropdown1" class = "tab-pane fade">
  <h3> Section III </h3>
  <p> Content ... </p>
```

```
</div>
```

```
<div id = "dropdown2" class = "tab-pane fade">
```

```
<h3> Section IV </h3>
```

```
<p> Content ... </p>
```

```
</div>
```

```
</div>
```

The .nav class and the .nav-tab class are used with the tag that is used to create the tab list. The data-toggle = "tab" property is used for each item in the list. >, You can add tab content by using the .tab-content class with the <div> tag that contains the content of all tabs, and using the .tab-pane class with the <div> tag containing the tab content, and don't forget to add the id with each < div> to be referenced in the href = "# id" property in the tab list.

Navbar menu bar

The Bootstrap framework provides multiple menu bars, with class names and a little effort you can add and customize as needed.

The default menu bar

We will start by explaining the default toolbar, which does not contain any additional elements only menu items with the `` tag, can be added by adding the `.navbar` class, the `.navbar-default` class and the `role = "navigation"` property to the `<nav>` tag, as follows:

```
<nav class = "navbar navbar-default" role = "navigation">
  <div class = "navbar-header">
    <a class="navbar-brand" href="#"> Bootstrap 3 framework </a>
  </div>

  <div>
    <ul class = "nav navbar-nav">
      <li class = "active"> <a href="#"> Home </a> </li>
      <li> <a href="#"> About the author </a> </li>
      <li class = "dropdown">
        <a href="#" class="dropdown-toggle" data-toggle="dropdown">
          Framework components
          <b class = "caret"> </b>
        </a>
        <ul class = "dropdown-menu">
          <li> <a href="#"> drop-down menus </a> </li>
          <li> <a href="#"> Tabs </a> </li>
          <li> <a href="#"> Menu bars </a> </li>
          <li class = "divider"> </li>
          <li> <a href="#"> Buttons </a> </li>
```

```
<li class = "divider"> </li>
<li> <a href="#"> Templates </a> </li>
</ul>
</li>
</ul>
</div>
</nav>
```

Before we move on to talk about additional properties let's turn a little limp on the previous code. The .navbar-brand tag is an image or text of the site logo, which is distinct from the rest of the menu items. The .dropdown class used with the class is used to create a drop-down list as we already knew, and the .active class makes the element in which it is placed. Active item, the item selected.

Responsive menu bar

The default mode in the toolbar is suitable for large, medium, and tablet screens, but the menu bar can also be responsive to smaller screens by using the `.collapse` and `.navbar-collapse` categories to the `<nav>` tag, which will reduce the menu to a button in the case of screens. When you click the button, the menu will drop down with all its contents, as shown in the following figure:

The drop-down menu bar code is:

```
<nav class = "navbar navbar-default" role = "navigation">
  <div class = "navbar-header">

    <button type = "button" class = "navbar-toggle" data-toggle = "collapse" data-target = "#
example-navbar-collapse">
      <span class = "sr-only"> drop-down menu </span>
      <span class = "icon-bar"> </span>
      <span class = "icon-bar"> </span>
      <span class = "icon-bar"> </span>
    </button>
    <a class="navbar-brand" href="#"> Responsive bar </a>

  </div>

  <div class = "collapse navbar-collapse" id = "example-navbar-collapse">
    <ul class = "nav navbar-nav">
      <li class = "active"> <a href="#"> Home </a> </li>
      ...
    </ul>
  </div>
</nav>
```

I transferred you the part of the modifications, while the rest of the code as shown in the default tape code. As you can see, we used the .navbar-toggle class with the <button> button tag to show or hide the menu inverting the position it is in (if it is hidden and if it is hidden). In the previous image, we also add the data-target = "# id" property and refer to the ID of the <div> tag added in this code to contain the complete menu items and take the .collapse class and the .navbar-collapse class, which gives it the collapse property of the menu bar collapse. Small screens case.

Add other items in the menu bar

You can add different items to the menu bar, such as the Text, Buttons, Captions, Links, and Template tools in general, as needed. You may need to add a search box, add annotations, or appropriate links (such as social media links, or in-site links), as in The following format:

Here is the code of the previous figure:

```
<nav class = "navbar navbar-default" role = "navigation">
  <div class = "navbar-header">
    <a class="navbar-brand" href="#"> Add templates </a>
  </div>

  <div>
    <button type = "button" class = "btn btn-default navbar-btn navbar-right">
      <span class = "glyphicon glyphicon-download"> </span> Loading product
    </button>

    <form class = "navbar-form navbar-right" role = "search">
      <div class = "form-group">
        <input type = "text" class = "form-control" placeholder = "Search for ...">
      </div>
      <button type = "submit" class = "btn btn-default"> Search </button>
```

```

</form>

<p class = "navbar-text navbar-left" style = "text-align: left;"> This is text that can illustrate
anything </p>
<a href="#" class="navbar-link navbar-left"> This is a link </a>
</div>
</nav>

```

So the Bootstrap framework allows you to add many elements. For example, a pattern, a button, a hyperlink, and plain text have been added. You can also see in the previous figure that the elements were not in one position, but there were elements on the left and the other on the right. Menus, you can align items using the .navbar-left or .navbar-right class, but only if they contain the .navbar- * class and * symbol. .navbar-btn).

Fixed mobile menu bar

It is true that you may not be able to move it wherever you want but this menu moves when you move the mouse wheel or when you move the side slider, up and down, and the bar may be at the top or bottom of the page as needed. You can add this bar using the .navbar-fixed- * class with the <nav> tag and the * symbol. Replaced with top or bottom depending on where you want the menu bar to appear, as in the following image. You notice the top and bottom menu, but you can't try moving it because it's an image. ! You will find out when you apply this part of the article:

Here is the code of the previous figure:

```

<nav class = "navbar navbar-default navbar-fixed-top" role = "navigation">
...
</nav>

```

```
<nav class = "navbar navbar-default navbar-fixed-bottom" role =  
"navigation">
```

```
...
```

```
</nav>
```

Static menu bar

This type of menu bar is stable in the site, and the .navbar-fixed class cannot be added to it, and the .navbar-static- * class is used. * As mentioned above, the * symbol is replaced in the case of the top menu with top and the bottom menu with the bottom. The format will be the same as the previous ones, you can see it through the examples provided or you can apply it yourself.

Distinct formatting of the Bootstrap framework for the menu bar

The .navbar-inverse class is used to reverse the foreground color with the back color, black becomes white and white is black. This gives a distinctive shape to the list, as you can see in the following picture:

This is part of the code to reverse the menu bar colors:

```
<nav class = "navbar navbar-default navbar-inverse" role =  
"navigation">
```

```
<div class = "navbar-header">
```

```
...
```

```
</nav>
```

Lists

Lists `` are basic HTML, and in Bootstrap, you can create a list in an organized and simple way. Once you use the `.list-group` class with the `` tag and the `.list-group-item` class with the `` tag, you will create a great list format. Somewhat - because there are some classes and characteristics we will talk about in the following paragraphs of the article - not only that but there are components that you can add to the list, and the following code to create a list of four elements:

```
<ul class = "list-group">  
  <li class = "list-group-item"> Products </li>  
  <li class = "list-group-item"> General Services </li>  
  <li class = "list-group-item"> Top Customers </li>  
  <li class = "list-group-item"> Discounts </li>  
</ul>
```

You can also add the previous list to your site through the `<a>` tag, but it should include all `<a>` tags that are the list under the `<div>` tag that will take the `.list-group` class while the `<a>` tag will take the `.list-group-item`. This is the menu configuration code using the `<a>` tag. The result will be the same as in the previous picture:

```
<div class = "list-group">  
  <a class="list-group-item"> Products </a>  
  <a class="list-group-item"> Public Services </a>  
  <a class="list-group-item"> Major customers </a>  
  <a class="list-group-item"> Discounts </a>  
</div>
```

More formats and items in the list

There are some formats that you can add through the Bootstrap framework to the list, such as adding a title and placing text in the same list or adding a title to a group of items within the list. The `.list-group-item-heading` class is used with the `<h *>` tag. Used as the title of an item in the list, the `.list-group-item-text` class is used with the `<p>` tag to write the content of the menu item, and to label a group of items within the list, we use the `.active` text with the `<a>` tag, ie it is an item in the list but contains The `<h *>` tag used for addressing as in the following code contains the title of a set of items and the title and content of each element:

```
<div class = "list-group">
```

```
<a href="#" class="list-group-item active">
```

```
<h4 class = "list-group-item-heading">
```

```
  Products
```

```
</h4>
```

```
</a>
```

```
<a href="#" class="list-group-item">
```

```
<h4 class = "list-group-item-heading">
```

```
  foodstuffs
```

```
</h4>
```

```
<p class = "list-group-item-text">
```

```
  We produce many foodstuffs such as biscuits, juices ...
```

```
</p>
```

```
</a>
```

```
<a href="#" class="list-group-item">
```

```
<h4 class = "list-group-item-heading">
```

```
  sweets
```

```
</h4>
```

```
<p class = "list-group-item-text">
```



```
    Sweets are the most important products that distinguish us ...
  </p>
</a>
</div>
```

```
<div class = "list-group">
  <a href="#" class="list-group-item active">
    <h4 class = "list-group-item-heading">
      Imported products
    </h4>
  </a>
```

```
<a href="#" class="list-group-item">
  <h4 class = "list-group-item-heading">
    Sugar
  </h4>
  <p class = "list-group-item-text">
    We import the finest Brazilian sugar ...
  </p>
</a>
</div>
```

These are the outputs of the previous code, as you notice there is a label for a set of items in the list and an address in the same element with content describing the title in each element:

Add Badges Badges to the list

Badges are great additions that can be combined with menus, giving an impression of the contents of the menu or a reference to a specific item in the list that is distinct from the rest of the items as you notice in the following picture:

The code will add distinctive signals as follows:

```
...  
<li class = "list-group-item"> Top Customers </li>  
<li class = "list-group-item">  
  <span class = "badge"> New </span> Branches  
</li>  
<li class = "list-group-item"> Sales </li>  
<li class = "list-group-item">  
  <span class = "badge"> Limited time </span> Last season discounts  
</li>  
...
```

All we used is to tag the `` with the `.badge` class that calls the distinctive sign at the end of the list.

So we have finished mentioning some components of the bootstrap

Chapter IV

The tree menu

Bootstrap provides a good working environment to build web pages interfaces professionally and easy to use by visitors to the site, in addition to attractive designs comfortable for the eye to browse and navigate within the site.

Today we are going to build an expandable or collapse tree list within the site's sidebar, which is one of the key pillars when building existing sites in their page ranking on a tree structure.

example.gif

An add-on built by Pomazan Max can be used to build a tree list and be published on GitHub. This addendum is subject to the terms of the MIT Agreement through the use of jquery.treegrid.js and jquery.treegrid.css

We will now customize and build this tree.

First it is important to add the css files that we use to build the tree, which are our files Main file for the tree list jquery.treegrid.css and bootstrap files

```
<link rel = "stylesheet" href = "css / bootstrap.min.css">
```

```
<link rel = "stylesheet" href = "css / bootstrap-theme.min.css">
```

```
<link rel = "stylesheet" href = "css / main.css">
```

```
<link rel = "stylesheet" href = "css / jquery.treegrid.css">
```

We build a panel-group to ensure the three nodes inside.

The id, which has the accordion value, is a bootstrap identifier and is useful for switching between open or collapsed nodes

```
<div class = "panel-group" id = "accordion">  
</div>
```

We add a title for the side menu

<h3> Try Tree Gird Sidebar </h3>

To build the first node node1 we add the instructions where we added node1 as the title of the list underneath node1-1 ... and therefore included it under the name of the row panel-heading As for the list below it was built as a list where the list-group-item row is added to each element of the list The way we added a submenu under the menu item node1-3

01.JPG

In order to change the title format for each master node node1, node2, node3 by adding one of the panel-info rows, which gives blue panel-danger, red, panel-success and green

02.JPG

```
<div class = "panel panel-info">  
<div class = "panel-heading">  
<div class = "panel-title">  
<a data-toggle="collapse" data-parent="#accordion" href="#node1"> Node1 </a>  
</div>
```

```

</div>
<div id = "node1" class = "panel-collapse collapse in">
<div class = "panel-body">
<div class = "list-group-item"> Node1-1 </div>
<div class = "list-group-item"> Node1-2 </div>
<div class = "list-group-item">
<div class = "panel-title">
<a data-toggle="collapse" href="#node1-3">
Node1-3
<span class = "glyphicon glyphicon-chevron-down"> </span>
</a>
</div>
<div id = "node1-3" class = "panel-collapse collapse">
<ul class = "list-group">
<li class = "list-group-item"> node1-3-1 </li>
<li class = "list-group-item"> node1-3-2 </li>
<li class = "list-group-item"> node1-3-3 </li>
</ul>
</div>
</div>
</div>
</div>
</div>
</div>

```

In the same way we build nodes2 and node3

If you want to change any menu you can do this:

To add a new node and a new address is done by adding a new panel under panel-group

Adds a panel element that is added as list-group-item

Each internal node can add a link to be able to access a different page by placing the node name under the link code <a> node1-1

At the end of the html file we add the javascript files that we use in the bootstrap to build the tree list jquery.treegrid.js

```
<script src = "js / vendor / jquery-1.11.2.min.js"> </script>  
<script src = "js / vendor / bootstrap.min.js"> </script>  
<script src = "js / vendor / jquery.treegrid.js"> </script>  
<script src = "js / main.js"> </script>
```

Within the Main.js JavaScript file we build the function that opens and closes each node when clicked by adding the active row when the node is opened.

```
function openNode (evt, nodeName) {  
    // Declare all variables  
  
    var i, tablinks;  
  
    // Get all elements with class = "tablinks" and remove the class  
    "active"  
    tablinks = document.getElementsByClassName ("tablinks");  
    for (i = 0; i < tablinks.length; i++) {  
        tablinks [i] .className = tablinks [i] .className.replace ("active",  
        "");  
    }  
  
    // Show the current tab, and add an "active" class to the link that  
    opened the tab
```

```
document.getElementById (nodeName) .style.display = "block";  
evt.currentTarget.className + = "active";  
}
```

We add the child that starts the page loading and closing all nodes that were open ie returning all variables to the default state

```
$ (document) .ready (function () {  
  $ ('# accordion'). find ('. collapse'). collapse ('hide');  
  var $ myGroup = $ ('# accordion');  
  $ myGroup.on ('show.bs.collapse', '.collapse', function () {  
    $ myGroup.not ($ (this) .parents ()). find ('. collapse.in'). collapse  
('hide');  
    });  
}  
);
```

So we have created a dynamic collapsible and scalable tree menu that enables the user to easily view the contents of the site

Chapter V

Build a site

The use of frameworks is of great benefit to the developer, since the developer no longer has to think about recurring and possibly frivolous problems, problems that have already been solved by framework makers, for example, compatibility between browsers, support for different screen dimensions or sizes (This method or mode of development greatly accelerates the planning and development of the site.

The Bootstrap framework is renowned for its wide popularity, which makes maintaining and revising any developer code easy to modify by any other developer, while keeping in mind its strong online support and documentation.

The disadvantages of using frameworks arise in the fact that the page should bear the burden of the entire frame code (the sum of the excessive formatting), and even when a small part of the frame is used, on the other hand, the frameworks are an effective tool in building prototyping, or in Creating pages where the aesthetic design is secondary, as with administration pages, but if you want to create a specific design with specific specifications, the use of these frameworks may make the design process more difficult than it is to build the design from Zero point, however, is possible and not impossible.

The final project will be as follows:

final-project.thumb.png.4b859bddf7aa5aad

Note about using Bootstrap

There are several ways to work with Bootstrap formats, either without using LESS or using LESS.

Use Bootstrap without LESS

Beginners with Bootstrap are encouraged to download the compiled / compiled copy of Bootstrap and attach it to the project, then create an empty CSS file and associate it with the project after bootstrap.css:

```
<link href = "css / bootstrap.min.css" rel = "stylesheet">
```

```
<link href = "css / styles.css" rel = "stylesheet">
```

To change from the default bootstrap formats, you must type in the file style.css:

```
a {color: # 00ff00}
```

```
block {background-color: #dddddd}
```

The disadvantage of using the above method is obviously that the developer must manually search for the correct format when changing, and it will not be easy in most cases, since some parameters are applied to many selectors when modifying. The customization of the framework can help the developer somewhat, since they can translate all the developer changes, once, but if the developer wants to change a new parameter, they must update all field values to be translated again.

Use Bootstrap with LESS

This method works by making all Bootstrap variables stored in .less files, and the developer should work with these variables and translate them into .css files (manually or automatically) as necessary. In HTML, all you have to do is link the compiled CSS files, so this is the best and most flexible method.

There are many ways to compile or translate LESS files, and the developer has the freedom to choose. Bootstrap itself uses Grunt to compile fewer files. One of the options available is WinLess for Windows users, SimpLESS for Mac users, or Koala for Linux users. By accessing LESS files and waiting for any changes to occur, when the developer makes this change, the tool will compile and create the corresponding CSS file, so the developer does not have to manually compile or compile after each change. Save, to show the results directly On the site which is already translated (compiled) and compressed (compressed).

Build the project using Bootstrap

Initially, the project files will be constructed by:

Create a folder named Project Whitesquare-bootstrap.

Create two subfolders: src for source files, and www for final site files.

Create an empty folder with images and an empty file with index.html inside the www folder.

Download Bootstrap and copy the contents of the zip file to the www folder.

Download the Bootstrap source files, copy the less folder, and place it in the project's src folder.

Create two files next to the less / bootstrap folder, one named styles.less and the second named variables.less, to be used to modify the basic Bootstrap properties. This provides a speed for updating and modifying the format.

file-structure.thumb.png.74a5928806ace26

The next step will be to translate LESS files into CSS.

It will work on the WinLess tool, which is easy to use, all you have to do is choose 'Add folder' from them, and then specify the path of the folder containing the files LESS:

C: \ whitesquare-bootstrap \ src \ less

The previous folder contains a list of all files, the last two files will be selected: styles.less and variables.less, then right click and choose from the 'Select output file' drop-down menu and then select the path of the CSS files.

.. \ .. \ www \ css \ styles.css

.. \ .. \ www \ css \ variables.css

Then, when any modification is made to the LESS files, the translation will be re-compiled to get the CSS files compiled with the new modifications.

Notes

The structure and structure of the project is ready after the creation of the previous files, but the developer should consider the following things before starting the design:

How the images will be presented, how they will be distributed and divided across the page.

How components will be used.

What are the basic formats?

What is the final plan.

After answering the previous questions it is possible to proceed with the design.

Prepare site images

At this stage, the images that will be used on all pages and not related to the content will be processed. The images in the current project will be as follows:

Image showing site address (map):

images / map.png

Site Logo Images:

images / logo.png

images / footer-logo.png

Background pictures:

/images/bg.png

/images/h1-bg.png

Social icon images, divided into two images to use sprite with them to load the page faster:

/images/social.png

/images/social-small.png

Components

The difference between website design with Bootstrap and design using native tools is that Bootstrap introduces the concept of components. These components represent common parts of pre-prepared HTML with their format, sometimes using JavaScript, and can be used. Bootstrap components are the same, or these components can be reformatted, all they need is to change the values of the variables in Bootstrap so, but if you want more flexibility to change, the developer can always change and modify the HTML and CSS as they like, and return to the project. It can be seen, The following components will be needed:

In order to design and distribute columns, the grid system (row, col) will be used.

In order to perform the search field, which is a form (inline) type (form-inline - input-group - btn)

For navigation, the <nav> tag will be used with the navbar class.

For submenus, a group list will be used using the list-group class.

For the panel, half of the panel will be used.

For a large appearance board, the jumbotron class will be used.

For picture frames, thumbnail will be used.

All previous components appear in the Bootstrap component documentation.

Basic format

Bootstrap already has most of the required formats for the project, but they will only need to be modified when needed, which can be done by modifying the `src / less / variables.css` file.

First, some variables that are not set by default in Bootstrap will be added first, and for later use:

The following variable sets the font used in the project:

@ brand-font: 'Oswald', sans-serif;

The Bootstrap options will also be adjusted to match the project's vision, which will mostly be for colors:

```
/* gray background of the page */  
@ body-bg: # f8f8f8;  
/* blue background */  
@ brand-primary: # 29c5e6;  
/* background of panels */  
@ panel-bg: # f3f3f3;  
/* frame color of panels */  
@ panel-inner-border: # e7e7e7;  
/* remove rounding in blocks */  
@ border-radius-base: 0;  
/* primary buttons have blue background */
```

```
@ btn-primary-bg: @ brand-primary;  
/* if the screen width is more then 992px, then the container width is  
960px */  
  
@ container-md: 960px;  
/* if the screen width is more 1200px, then the container width is 960px  
again */  
  
@ container-lg: @ container-md;  
/* main font is Tahoma */  
  
@ font-family-base: Tahoma, sans-serif;  
/* base font size */  
  
@ font-size-base: 12px;  
/* main color of text */  
  
@ text-color: # 8f8f8f;  
/* gray background of text fields */  
  
@ input-bg: @ panel-bg;  
/* gray frame of text fields */  
  
@ input-border: @ panel-inner-border;  
/* gray color of the text in the fields */  
  
@ input-color: # b2b2b2;
```

Once the variables have been completed, the formatting of the project will be started in the styles.less file, but you must first import the public Bootstrap file and the variables file:

```
@import "bootstrap / bootstrap.less";  
@import "variables.less";
```

Note that not all formats (which are set by the frame itself) can be changed using variables, but must be changed manually:

```
p {  
  margin: 20px 0;  
}
```

```
.form-control {  
  box-shadow: none;  
}
```

```
.btn {  
  font-family: @ brand-font;  
}
```

```
body {  
  border-top: 5px solid # 7e7e7e;  
  background-image: url (../ images / bg.png);  
}
```

The preceding lines will remove the shadow of the form elements, define a special line of text within the button, and add a background image to the entire page and a top border.

Where the layouts are placed will no longer be listed, they will always look like this: variables in the variables.less file, and all custom formats will be in the styles.less file.

HTML structure

A site plan usually starts with an HTML skeleton, which looks like this:

```
<!DOCTYPE html>

<html>

  <head>

    <title> Bootstrap 3 page layout </title>

    <meta name = "viewport" content = "width = device-width, initial-
scale = 1.0">

    <link href = "css / styles.css" rel = "stylesheet">

    <!-- [if lt IE 9]>

      <script src =
"https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"> </script>

      <script src =
"https://oss.maxcdn.com/libs/respond.js/1.3.0/respond.min.js">
</script>

    <!-- [endif] -->

  </head>

  <body>

  </body>

</html>
```

In the previous lines, the main HTML5 document was constructed. The title tag referred to the page title, 'Bootstrap 3 page layout'. In the <meta> tag, the page width on mobile devices was equal to the screen width, and the zoom level was adjusted. (zoom level) In the first loading of the page, and then linked to the format sheet (stylesheet), and for the browser Internet

Explorer (before the ninth version) was written a script that allows the layout of the page properly.

Page Layout

The page layout consists of two parts: the main content container, which is centered on the screen and the footer, and the main container consists of two columns: the main content, the sidebar (sidebar), and comes from the top of the header, and the navigation bar (nav) , And heading (heading).

page-layout.thumb.png.59c6ea858c0d09b690

The following code will be added to the trunk of the page:

```
<body>
  <div class = "wrapper container">
    <header> </header>
    <nav> </nav>
    <div class = "heading"> </div>
    <div class = "row">
      <aside class = "col-md-7"> </aside>
      <section class = "col-md-17"> </section>
    </div>
  </div>
  <footer> </footer>
</body>
```

The previous code provides the general division of columns, which is contained within a class called row, and the column classes begin with the col- prefix, and then the screen size is one of four (xs, sm, md, lg). Ends with column width value.

Columns can define together the values of different screen types, and they represent the width of the column for different screen sizes, for example:

class = "col-xs-12 col-md-8"

Class col-xs-12 will be applied with large screens, and col-md-8 will be applied to small screens. These classes will be applied to the width of larger screens equal to the value of the selected item. Therefore, applying col-md-* to the element will not affect not only medium-sized monitors but also large monitors, when no col-lg-* class is present.

In the previous code, col-md-7 and col-md-17 varieties were used, which indicate that the <aside> block will have a 7-column column width in the intermediate screens, and the <section> block will have a 17-column width, relative to the parent container, the total column capacity in Bootstrap is 12, but the number has been increased (to double) in order to have more flexibility in distribution, but in general the use of 12 columns is suitable for most designs.

In the following code, some padding will be added to fit the format, to the Wrapper class, and the header of the page:

```
body {  
  ...  
  
  .wrapper {  
    padding: 0 0 50px 0;  
  }  
  
  header {  
    padding: 20px 0;  
  }  
}
```

In the previous code, the preprocessor was used in formatting the format. This allows us to write nested and non-repetitive formats, which will be drafted after translation into:

body .wrapper {...}

body header {...}

This method of setup allows you to see the correct HTML structures within CSS, and provides a sort and organization of CSS code writing.

Conclusion

In this section, Bootstrap is introduced, the work environment is set up, and the structure of the project files is built. In the second part, each part of the page will be designed.

Chapter VI

Build a site

Logo

logo.thumb.png.0f486ddbdef3defa74fdbf0c8

The logo will be added to the header as follows:

```
<header>
```

```
  <a href="/"> <img src = "images / logo.png" alt = "Logo"> </a>
```

```
</header>
```

The logo image will be inserted without adding any layouts.

Search form

search.thumb.png.d61824dc3af5dba9200f493

Some Bootstrap components will be used to create the search form. A form will be created within the line in the header of the page. This form will also be shifted to the right. This field will have the form-control and label.

The form will contain the input groups, whose task is to remove the gap between the input text field and the button, as if it merges them as a single element, so the input-group class will be used, and the input element will use the form -control For the button section, the input-group-btn class will be used.

The btn-primary item will then be added to the button, which will give the button a prominent color and border.

```
<header>
```

```
...
```

```
<form name = "search" action = "#" method = "get" class = "form-  
inline form-search pull-right">
```

```
<div class = "input-group">
```

```
<input class = "form-control" id = "searchInput" type = "text"  
name = "search" placeholder = "Search">
```

```
<div class = "input-group-btn">
```

```
<button type = "submit" class = "btn btn-primary"> GO </button>
```

```
</div>
```

```
</div>
```

```
</form>
```

```
</header>
```

In the next step, the search box width will be set to 200px:

```
body {
```

```
...
```

```
.wrapper {
```

```
...
```

```
header {
```

```
...
```

```
.form-search {
```

width: 200px;

}

}

}

}

Navigation bar

navbar.thumb.png.ca700d9ca1245d2d5e8af8c

The navigation components, which contain a list of links, will be used, and the navbar-nav class will be used for the navigation bar, which applies a special navigation bar format.

```
<nav class = "navbar navbar-default">
  <ul class = "nav navbar-nav">
    <li> <a href="/home/"> Home </a> </li>
    <li class = "active"> <a href="/about/"> About us </a> </li>
    <li> <a href="/services/"> Services </a> </li>
    <li> <a href="/partners/"> Partners </a> </li>
    <li> <a href="/customers/"> Customers </a> </li>
    <li> <a href="/projects/"> Projects </a> </li>
    <li> <a href="/careers/"> Careers </a> </li>
    <li> <a href="/contact/"> Contact </a> </li>
  </ul>
</nav>
```

These variables will also be added in order to make the navigation bar format more compatible with the page.

```
/* navigation menu height */
@ navbar-height: 37px;

/* additional paddings */
@ nav-link-padding: 10px 30px;

/* background for menu items */
@ navbar-default-bg: @ panel-bg;
```

```
/* text color in the menu items */  
@ navbar-default-link-color: # b2b2b2;  
  
/* for the mouse hover - the same color */  
@ navbar-default-link-hover-color: @ navbar-default-link-color;  
  
/* background of the active menu item */  
@ navbar-default-link-active-bg: @ brand-primary;  
  
/* text color of the active menu item */  
@ navbar-default-link-active-color: #fff;
```

The following modifications will also be added to the project's format file, to customize the font used for the navigation bar, by uppercase and adjusting the font type and size.

```
body {  
  ...  
  .wrapper {  
    ...  
    .navbar a {  
      text-transform: uppercase;  
      font: 14px @ brand-font;  
    }  
  }  
}
```

Page Header

page-header.thumb.png.57833ed395a60d2cde

The heading heading will be used with the page title:

```
<div class = "heading">  
  <h1> About us </h1>  
</div>
```

In the following format:

```
body {  
  ...  
  
  .wrapper {  
    ...  
  
    .heading {  
      height: 40px;  
      background: transparent url (../ images / h1-bg.png);  
      margin: 30px 0;  
      padding-left: 20px;  
  
      h1 {  
        display: inline-block;
```



```
color: # 7e7e7e;  
font: normal 40px / 40px 'Oswald', sans-serif;  
background: url (../ images / bg.png);  
margin: 0;  
padding: 0 10px;  
text-transform: uppercase;  
}  
}  
}  
}
```

In the previous code we set a background image for the title (h1), and the font formatting, to show the title in the previous format.

Submenu

aside.thumb.png.5fc205c8848c6fdbf7e586f5

The navigation component will not be used with the submenu, as it is not suitable for formatting. Instead, the group list component will be used, and each component of this component will own the list-group-item class. Note that the submenu should be placed inside the aside tag:

```
<aside class = "col-md-7">
  <ul class = "list-group submenu">
    <li class = "list-group-item active"> Lorem ipsum </li>
    <li class = "list-group-item"> <a href="/donec/"> Donec tincidunt laoreet </a> </li>
    <li class = "list-group-item"> <a href="/vestibulum/"> Vestibulum elit </a> </li>
    <li class = "list-group-item"> <a href="/etiam/"> Etiam pharetra </a> </li>
    <li class = "list-group-item"> <a href="/phasellus/"> Phasellus placerat </a> </li>
    <li class = "list-group-item"> <a href="/cras/"> Cras et nisi vitae odio </a> </li>
  </ul>
</aside>
```

Component settings show that all grouped lists use the background and boundaries of the panel component:

@ list-group-bg: @ panel-bg;

@ list-group-border: @ panel-inner-border;

The following formats will then be applied to the submenu:

body {

...

```
.wrapper {  
  ...  
  
  .submenu {  
    margin-bottom: 30px;  
  
    li {  
      display: list-item;  
      font: 300 14px @ brand-font;  
      list-style-position: inside;  
      list-style-type: square;  
      padding: 10px;  
      text-transform: uppercase;  
  
      & .active {  
        color: @ brand-primary;  
      }  
  
      a {  
        color: @ text-color;  
        text-decoration: none;  
  
        &: hover {  
          color: @ text-color;  
        }  
      }  
    }  
  }  
}
```

A bottom margin has been added using margin-bottom, the font is lightened, and the menu items are formatted using the list-style-type

property to be square, and for the links the color and letters are set to uppercase. (ampersand) is part of the LESS syntax, which will be replaced by the parent selector.

Sidebar

At the bottom of the submenu, a picture indicating the address of the employment office will be added.

sidebar.thumb.png.70fb2a9f24b00c19c3363c

The panel component of the Bootstrap component will be used, using its sub-classes, panel-primary, for coloring the title. Image of the sitemap, in order to have a responsive different screen measurements (responsiveness).

```
<aside class = "col-md-7">
...
<div class = "panel panel-primary">
  <div class = "panel-heading"> Our offices </div>
  <div class = "panel-body">
    <img src = "images / map.png" class = "img-responsive" alt = "Our
offices">
  </div>
</div>
</aside>
```

The background of the panel component has already been set in the Bootstrap properties via the panel-bg variable, and now what will be done is to change its border color, and the gray color previously assigned to the @panel-inner-border variable will be used:

```
@ panel-primary-border: @ panel-inner-border;
```

You will also need to change some default panel formatting, and these changes cannot be changed by the variables:

```
.panel {  
  box-shadow: none;  
  .panel-heading {  
    font: 14px @ brand-font;  
    text-transform: uppercase;  
    padding: 10px;  
  }  
  
  .panel-body {  
    padding: 10px;  
  }  
}
```

In the previous code, the shadow of the panel, padding, and line adjustment were removed.

Quotation

We will start working on content design, and in the beginning we will add a quote:

quotation.thumb.png.718ada27d3033e82b8f9

This part is very similar to the Jumbotron component, which will be used here, by adding it inside the content column (col-md-17):

```
<section class = "col-md-17">
  <div class = "jumbotron">
    <blockquote>
      <p>
        "Quisque in enim velit, at dignissim est. Nulla ul corper, dolor ac pellentesque placerat,
        justo tellus gravida erat, vel porttitor libero erat."
      </p>
      <small> John Doe, Lorem Ipsum </small>
    </blockquote>
  </div>
</section>
```

Using the jumbotron component variables, the white color of the font will be set, and the background color will be set to the @ brand-primary variable value of # 29c5e6:

@ jumbotron-bg: @ brand-primary;

@ jumbotron-color: #fff;

Some formatting will also be added:

```
body {
  ...
  .wrapper {
    ...
    .jumbotron {
      border-radius: 0;
      padding: 0;
      margin: 0;
```

```
blockquote {  
border-left: none;
```

```
p {  
font: 300 italic 33px @ brand-font;  
text-transform: uppercase;  
margin-bottom: 0;  
}
```

```
small {  
text-align: right;  
color: # 1D8EA6;  
font: 300 20px @ brand-font;
```

```
&: before {  
content: ";  
}  
}  
}  
}  
}  
}
```

The rounded corners have been removed using the border-radius feature, and the line is formatted to match the rest of the design.

Main content

main-content.thumb.png.12a4009bf6762120b

All the necessary text formats have already been completed, so all you need to do is add three paragraphs with the text:

<p> Lorem ipsum dolor sit amet ... </p>

<p> Donec vel nisl nibh ... </p>

<p> Donec vel nisl nibh ... </p>

In the next step, two images will be added, below the previous written content, so two columns will be used:

<div class = "row">

<div class = "col-md-12">

</div>

<div class = "col-md-12">

</div>

</div>

The thumbnail class gives a nice image format, and without adding more formatting, only what will be done is to adjust the padding and border color:

@ thumbnail-padding: 1px;

@ thumbnail-border: # c9c9c9;

Staff section

our-team.thumb.png.a2b672d8313763d999e5d

First, the header of this section will be added:

<h2> Our team </h2>

In the following format:

```
body {  
  ...  
  .wrapper {  
    ...  
    h2 {  
      background: none repeat scroll 0 0 # 29C5E6;  
      color: #fff;  
      font: 300 30px @ brand-font;  
      padding: 0 10px;  
      text-transform: uppercase;  
    }  
  }  
}
```

A div will be added with the team class, which contains the team cards, since each card is a column with a width equal to four columns of the grid system, all of which have one column offset except the first card of each row, and this offset is applied using the col-sm- offset-1, each card consists of an image and job description for each team member.

```
<div class = "team">
  <div class = "row">
    <div class = "col col-sm-4">
      <img src = "images / team / Doe.jpg" alt = "John Doe" class = "thumbnail">
      <div class = "caption">
        <h3> John Doe </h3>
        <p> ceo </p>
      </div>
    </div>
    <div class = "col col-sm-4 col-sm-offset-1">
      <img src = "images / team / Pittsley.jpg" alt = "Saundra Pittsley" class = "thumbnail">
      <div class = "caption">
        <h3> Saundra Pittsley </h3>
        <p> team leader </p>
      </div>
    </div>
    ...
  </div>
  <div class = "row">
    <div class = "col col-sm-4">
      <img src = "images / team / Nobriga.jpg" alt = "Ericka Nobriga" class = "thumbnail">
      <div class = "caption">
        <h3> Ericka Nobriga </h3>
        <p> art director </p>
      </div>
    </div>
    <div class = "col col-sm-4 col-sm-offset-1">
      <img src = "images / team / Rousselle.jpg" alt = "Cody Rousselle" class = "thumbnail">
      <div class = "caption">
        <h3> Cody Rousselle </h3>
        <p> senior ui designer </p>
      </div>
    </div>
    ...
  </div>
  ...
</div>
```

</div>

</div>

The following format will be added to the format file to match the previous content to the design:

```
body {  
  ...  
  .wrapper {  
    ...  
    .team {  
      .row {  
        margin-top: 20px;  
  
        .col {  
          white-space: nowrap;  
  
          .thumbnail {  
            margin-bottom: 5px;  
          }  
        }  
  
        .col-sm-offset-1 {  
          margin-left: 3.7%;  
        }  
  
        .caption {  
          h3 {  
            font: 300 16px @ brand-font;  
            margin: 0;  
          }  
  
          p {
```

```
font: 300 14px @ brand-font;
color: @ brand-primary;
margin: 0;
}
}
}
}
}
}
```

The col-sm-offset-1 code was modified in the previous code, as its margin is rather wide, and is therefore set at 3.7%.

Page tail design

The tail has four main parts:

Twitter feed

Site map

Social links

Logo with copyright text

footer.thumb.png.febafe0b8c422b9bd3da8af

Initially create the main container for the page footer, and the columns for each part:

```
<footer>
  <div class = "container">
    <div class = "row">
      <div class = "col-md-8 col-xs-12 twitter"> </div>
```

```
<div class = "col-md-4 col-xs-12 sitemap"> </div>
<div class = "clearfix visible-sm visible-xs"> </div>
<div class = "col-md-6 col-xs-12 social"> </div>
<div class = "col-md-6 col-xs-12 footer-logo"> </div>
</div>
</div>
</footer>
```

The footer will be formatted as follows:

```
footer {
  background: # 7e7e7e;
  color: #dbdbdb;
  font-size: 11px;
  overflow: hidden;

  .container {
    height: 110px;
    padding: 10px 0;
  }
}
```

The footer tag appears on the entire width of the screen, while the inner container appears in the middle. To align the elements within the footer, the column system is used.

Twitter feed

twitter-feed.thumb.png.c6414cf303b3501a1

The tensioning feed formulation is as follows:

```
<div class = "col-md-8 col-xs-12 twitter">  
  <h3> Twitter feed </h3>  
  <time datetime = "2015-03-03"> <a href="#"> 03 mar </a> </time>  
  <p> In ultricies pellentesque massa a porta. Aliquam ipsum enim, hendrerit ut porta nec,  
  ullamcorper et nulla. In eget mi dui, sit amet scelerisque nunc. Aenean aug </p>  
</div>
```

The format is as follows:

```
body {  
  ...  
  footer {  
    ...  
    .container {  
      ...  
      h3 {  
        border-bottom: 1px solid # 919191;  
        color: #ffffff;  
        font-size: 14px;  
        line-height: 21px;  
        font-family: @ brand-font;  
        margin: 0 0 10px;  
        text-transform: uppercase;  
      }  
  
      p {
```



```
margin: 5px 0;
}

.twitter {
  p {
    padding-right: 15px;
  }

  time a {
    color: # b4aeae;
    text-decoration: underline;
  }
}
}
```

The format of all the headings of the page is tailored to the font and margins, the use of uppercase letters, using the text-transform property, and for the link displaying the date, the color and the underlining are set.

Sitemap

A sitemap consists of two equal columns with links:

sitemap.thumb.png.a0241847d67e80a35a0ca4

```
<div class = "col-md-4 col-xs-12 sitemap">
  <h3> Sitemap </h3>
  <div class = "row">
    <div class = "col-md-12">
      <a href="/home/"> Home </a>
      <a href="/about/"> About </a>
      <a href="/services/"> Services </a>
    </div>
    <div class = "col-md-12">
      <a href="/partners/"> Partners </a>
      <a href="/customers/"> Support </a>
      <a href="/contact/"> Contact </a>
    </div>
  </div>
</div>
```

The following format, which is related to the color, font and margin of the links, will be applied:

```
body {
  ...
  footer {
    ...
    .container {
      ...
    }
  }
}
```

```
a {  
  color: #dbdbdb;  
}  
  
.sitemap a {  
  display: block;  
  font-size: 12px;  
  margin-bottom: 5px;  
}  
}  
}  
}
```

Social icons

social-buttons.thumb.png.721c8749be8cbec

All links (buttons) will be placed within a section and in the social category:

```
<div class = "col-md-4 col-xs-12 social">  
  <h3> Social networks </h3>  
  <a href="http://twitter.com/" class="social-icon twitter"> </a>  
  <a href="http://facebook.com/" class="social-icon facebook"> </a>  
  <a href="http://plus.google.com/" class="social-icon google-plus"> </a>  
  <a href="http://vimeo.com/" class="social-icon-small vimeo"> </a>  
  <a href="http://youtube.com/" class="social-icon-small youtube"> </a>  
  <a href="http://flickr.com/" class="social-icon-small flickr"> </a>  
  <a href="http://instagram.com/" class="social-icon-small instagram"> </a>  
  <a href="/rss/" class="social-icon-small rss"> </a>  
</div>
```

To coordinate links within the footer, the following will be applied:

```
body {  
  ...  
  footer {  
    ...  
    .container {  
      ...  
      .social {  
        .social-icon {  
          width: 30px;  
          height: 30px;  
          background: url (../ images / social.png) no-repeat;  
          display: inline-block;
```

```

    margin-right: 10px;
}

.social-icon-small {
    width: 16px;
    height: 16px;
    background: url (../ images / social-small.png) no-repeat;
    display: inline-block;
    margin: 5px 6px 0 0;
}

.twitter {background-position: 0 0; }
.facebook {background-position: -30px 0; }
.google-plus {background-position: -60px 0; }
.vimeo {background-position: 0 0; }
.youtube {background-position: -16px 0; }
.flickr {background-position: -32px 0; }
.instagram {background-position: -48px 0; }
.rss {background-position: -64px 0; }
}
}
}
}

```

In the previous code, a method known as sprites, a common method used to speed up page loading, was used by combining a set of images into a single image, so the browser would request one image instead of several images, ie one request instead of several requests, in the project. The current social icons are divided into two images: social.png and social-small.png, each using a background using the background property and positioning each background using the background-position feature.

Copyright text

An image with a link to the logo, and a paragraph below it, will be used with the copyright text.

copyright.thumb.png.5892f2729f5fac259046

The wording will be as follows:

```
<div class = "col-md-8 col-xs-12 footer-logo">  
  <a href="/"> <img src = "images / footer-logo.png" alt = "Whitesquare logo"> </a>  
  <p> Copyright © 2015 Whitesquare. </p>  
</div>
```

This section will be formatted similar to the previous one, with only one difference, the offset to the right:

```
body {  
  ...  
  .footer {  
    ...  
    .container {  
      ...  
      .footer-logo {  
        float: right;  
        margin-top: 20px;  
        font-size: 10px;  
        text-align: right;  
  
        a {  
          text-decoration: underline;
```

}
}
}
}
}