

Introduction

- Bootstrap is the currently most popular open source framework for developing **responsive**, mobile first Web application and websites.
- It uses HTML5, CSS3, and JavaScript and jQuery.
- Bootstrap helps you kick start the development of webapps and websites.
- Bootstrap contains HTML and CSS-based design templates for text, forms, buttons, navigation and other components.

Major Features of Bootstrap:

- **Built-in Support** for layout, grids, fluid grids, and responsive designs.
- **Pre-built CSS:** Contains global CSS classes for typography, tables, grids, forms, buttons, images, and more
- **Components:** Contains lots of reusable components including Icons, Dropdowns, Navbars, Breadcrumbs, Popovers, Alerts, and many more
- **JavaScript Plugins:** Contains lots of custom jQuery plugins. You can include them all or one by one
- **Customizable Components:** We can customize Bootstrap's components with LESS variables and jQuery plugins to create our own version.

History:

It's developed by Mark Otto and Jacob Thornton at **Twitter** as a framework to encourage consistency across internal tools. It's also referred as Twitter Bootstrap.

Its first version was released in Aug 2011 and because of the features it provided, in June 2014 Bootstrap was the No.1 project on GitHub.

Advantage of Bootstrap Framework

- **Mobile first approach** – Bootstrap 3, framework consists of Mobile first styles throughout the entire library instead them of in separate files.
- **Browser Support** – It is supported by all popular browsers.



- **Easy to get started** – With just the knowledge of HTML and CSS anyone can get started with Bootstrap. Also the Bootstrap official site has a good documentation.

- **Responsive design** – Bootstrap's responsive CSS adjusts to Desktops, Tablets and Mobiles.



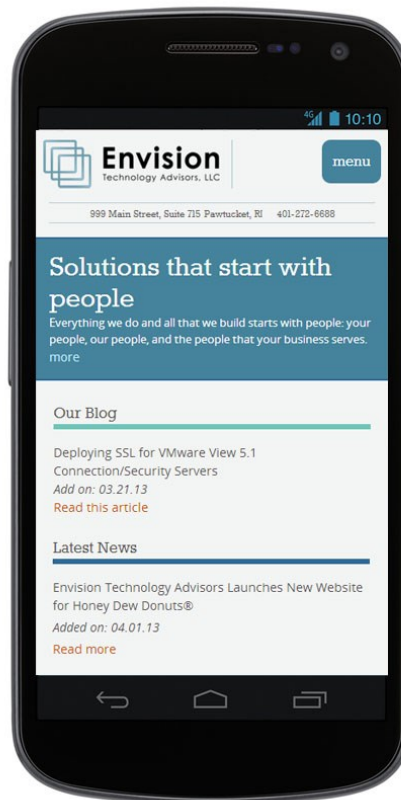
- Provides a clean and uniform solution for building an interface for developers.
- It contains beautiful and functional built-in components which are easy to customize.
- It also provides web based customization.
- And best of all it is an open source

Responsive VS Non-Responsive

The “do nothing” approach to mobile device support

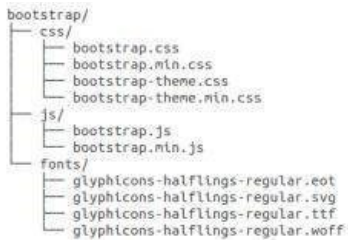


A “responsive” website with an optimized layout & experience



Environment Setup

Once the compiled version Bootstrap is downloaded, extract the ZIP file, and you will see the following file/directory structure –



As you can see, there are compiled CSS and JS (bootstrap.*), as well as compiled and minified CSS and JS (bootstrap.min.*). Fonts from Glyphicons are included, as it is the optional Bootstrap theme.

Bootstrap CDN

If you don't want to download and host Bootstrap yourself, you can include it from a CDN (Content Delivery Network).

MaxCDN provides CDN support for Bootstrap's CSS and JavaScript. You must also include jQuery

```
<!-- Latest compiled and minified CSS -->
<link rel="stylesheet" href="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css">

<!-- jQuery library -->
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.2/jquery.min.js"></script>

<!-- Latest compiled JavaScript -->
<script src="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js"></script>
```

Create First Web Page With Bootstrap

1. Add the HTML5 doctype

Bootstrap uses HTML elements and CSS properties that require the HTML5 doctype.

Always include the HTML5 doctype at the beginning of the page, along with the lang attribute and the correct character set:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
  </head>
</html>
```

2. Bootstrap 3 is mobile-first

Bootstrap 3 is designed to be responsive to mobile devices. Mobile-first styles are part of the core framework.

To ensure proper rendering and touch zooming, add the following `<meta>` tag inside the `<head>` element:

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

The `width=device-width` part sets the width of the page to follow the screen-width of the device (which will vary depending on the device).

The `initial-scale=1` part sets the initial zoom level when the page is first loaded by the browser.

You need to add the **viewport meta tag** to the `<head>` element, to ensure proper rendering and touch zooming on mobile devices.

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

- `width` property controls the width of the device. Setting it to `device-width` will make sure that it is rendered across various devices (mobiles, desktops, tablets...) properly.
- `initial-scale=1.0` ensures that when loaded, your web page will be rendered at a 1:1 scale, and no zooming will be applied out of the box.

```
<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">
```

Add **`user-scalable=no`** to the **`content`** attribute to disable zooming capabilities on mobile devices as shown below.

Normally **`maximum-scale=1.0`** is used along with **`user-scalable=no`**. This may give users an experience more like a native app, hence Bootstrap doesn't recommend using this attribute.

3. Containers

Bootstrap also requires a containing element to wrap site contents.

There are two container classes to choose from:

1. The `.container` class provides a responsive **fixed width container**
2. The `.container-fluid` class provides a **full width container**, spanning the entire width of the viewport

Note: Containers are not nestable (you cannot put a container inside another container).

Two Basic Bootstrap Pages

The following example shows the code for a basic Bootstrap page (with a responsive fixed width container):

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Bootstrap Example</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.0/jquery.min.js"></script>
  <script src="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js"></script>
</head>
<body>

<div class="container">
  <h1>My First Bootstrap Page</h1>
  <p>This is some text.</p>
</div>

</body>
</html>
```

BootStrap Example

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Bootstrap Example</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.2/jquery.min.js"></script>
  <script src="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js"></script>
</head>
<body>

<div class="container">
  <div class="jumbotron">
    <h1>My First Bootstrap Page</h1>
    <p>Resize this responsive page to see the effect!</p>
  </div>
  <div class="row">
    <div class="col-sm-4">
      <h3>Column 1</h3>
      <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit...</p>
      <p>Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris...</p>
    </div>
    <div class="col-sm-4">
      <h3>Column 2</h3>
      <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit...</p>
      <p>Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris...</p>
    </div>
    <div class="col-sm-4">
      <h3>Column 3</h3>
      <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit...</p>
    </div>
  </div>
</div>
```

```
<p>Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris...</p>
</div>
</div>
</div>

</body>
</html>
```

Bootstrap Grid

What is Grid

In graphic design, a grid is a structure (usually two-dimensional) made up of a series of intersecting straight (vertical, horizontal) lines used to structure the content. It is widely used to design layout and content structure in print design. In web design, it is a very effective method to create a consistent layout rapidly and effectively using HTML and CSS.

- In web design, a grid is a very effective method to create a consistent layout rapidly and effectively using HTML and CSS.
- Bootstrap includes a responsive, mobile first [fluid] grid system that appropriately scales up to 12 columns as the device or viewport size increases.
- It includes predefined classes for easy layout options, as well as powerful mixins for generating more semantic layouts.

Grid Rules

Grid systems are used for creating page layouts through a series of rows and columns that house your content. Here's how the Bootstrap grid system works –

- Rows must be placed within a **.container** class for proper alignment and padding.
- Use rows to create horizontal groups of columns.
- Content should be placed within the columns, and only columns may be the immediate children of rows.
- Predefined grid classes like **.row** and **.col-xs-4** are available for quickly making grid layouts. LESS mixins can also be used for more semantic layouts.
- Columns create gutters (gaps between column content) via padding. That padding is offset in rows for the first and the last column via negative margin on **.rows**.
- Grid columns are created by specifying the number of twelve available columns you wish to span. For example, three equal columns would use three. **col-xs-4**.

Basic Grid Structure

Following is basic structure of Bootstrap grid –

```
<div class = "container">
  <div class = "row">
    <div class = "col-*-*"></div>
```

```

    <div class = "col-*-*"></div>
  </div>

  <div class = "row">...</div>
</div>
<div class = "container">
  ....
</div>

```

Grid options

The following table summarizes aspects of how Bootstrap grid system works across multiple devices –

	Extra small devices Phones (<768px)	Small devices Tablets (≥768px)	Medium devices Desktops (≥992px)	Large devices Desktops (≥1200px)
Grid behavior	Horizontal at all times	Collapsed to start, horizontal above breakpoints	Collapsed to start, horizontal above breakpoints	Collapsed to start, horizontal above breakpoints
Max container width	None (auto)	750px	970px	1170px
Class prefix	.col-xs-	.col-sm-	.col-md-	.col-lg-
# of columns	12	12	12	12
Max column width	Auto	60px	78px	95px
Gutter width	30px (15px on each side of a column)	30px (15px on each side of a column)	30px (15px on each side of a column)	30px (15px on each side of a column)
Nestable	Yes	Yes	Yes	Yes
Offsets	Yes	Yes	Yes	Yes
Column ordering	Yes	Yes	Yes	Yes

Typography

Bootstrap uses Helvetica Neue, Helvetica, Arial, and sans-serif in its default font stack. Using typography feature of Bootstrap you can create headings, paragraphs, lists and other inline elements.

Headings

All HTML headings (h1 to h6) are styled in Bootstrap. An example is shown below –

```

<h1>I'm Heading1 h1</h1>
<h2>I'm Heading2 h2</h2>
<h3>I'm Heading3 h3</h3>
<h4>I'm Heading4 h4</h4>
<h5>I'm Heading5 h5</h5>

```

```
<h6>I'm Heading6 h6</h6>
```

Inline Subheadings

To add an inline subheading to any of the headings, simply add `<small>` around any of the elements or add `.small` class and you will get smaller text in a lighter color as shown in the example below –

```
<h1>I'm Heading1 h1. <small>I'm secondary Heading1 h1</small></h1>
<h2>I'm Heading2 h2. <small>I'm secondary Heading2 h2</small></h2>
<h3>I'm Heading3 h3. <small>I'm secondary Heading3 h3</small></h3>
<h4>I'm Heading4 h4. <small>I'm secondary Heading4 h4</small></h4>
<h5>I'm Heading5 h5. <small>I'm secondary Heading5 h5</small></h5>
<h6>I'm Heading6 h6. <small>I'm secondary Heading1 h6</small></h6>
```

Lead Body Copy

To add some emphasis to a paragraph, add `class = "lead"`. This will give you a larger font size, lighter weight, and a taller line height as in the following example –

```
<h2>Lead Example</h2>
<p class = "lead">This is an example paragraph demonstrating
the use of lead body copy. This is an example paragraph
demonstrating the use of lead body copy.This is an example
paragraph demonstrating the use of lead body copy.This is an
example paragraph demonstrating the use of lead body copy.
This is an example paragraph demonstrating the use of lead body copy.</p>
```

Emphasis

HTML's default emphasis tags such as `<small>` sets text at 85% the size of the parent, `` emphasizes a text with heavier font-weight, and `` emphasizes a text in italics.

Bootstrap offers a few classes that can be used to provide emphasis on texts as seen in the following example –

```
<small>This content is within tag</small><br>
<strong>This content is within tag</strong><br>
<em>This content is within tag and is rendered as italics</em><br>

<p class = "text-left">Left aligned text.</p>
<p class = "text-center">Center aligned text.</p>
<p class = "text-right">Right aligned text.</p>
<p class = "text-muted">This content is muted</p>
<p class = "text-primary">This content carries a primary class</p>
<p class = "text-success">This content carries a success class</p>
<p class = "text-info">This content carries a info class</p>
<p class = "text-warning">This content carries a warning class</p>
<p class = "text-danger">This content carries a danger class</p>
```


Abbreviations

The HTML <abbr> element provides markup for abbreviations or acronyms, like WWW or HTTP. Bootstrap styles <abbr> elements with a light dotted border along the bottom and reveals the full text on hover (as long as you add that text to the <abbr> title attribute). To get a slightly smaller font size add .initialism to <abbr>.

```
<abbr title = "World Wide Web">WWW</abbr><br>
<abbr title = "Real Simple Syndication" class = "initialism">RSS</abbr>
```

Addresses

Using <address> tag you can display the contact information on your web page. Since the <address> defaults to display: block; you'll need to use

Tags to add line breaks to the enclosed address text.

```
<address>
  <strong>Some Company, Inc.</strong><br>
  007 street<br>
  Some City, State XXXXX<br>
  <abbr title = "Phone">P:</abbr> (123) 456-7890
</address>

<address>
  <strong>Full Name</strong><br>
  <a href = "mailto:#">mailto@somedomain.com</a>
</address>
```

Blockquotes

You can use the default <blockquote> around any HTML text. Other options include, adding a <small> tag for identifying the source of the quote and right-aligning the blockquote using class .pull-right. The following example demonstrates all these features –

```
<blockquote>
  <p>This is a default blockquote example. This is a default
    blockquote example. This is a default blockquote
    example.This is a default blockquote example. This is a
    default blockquote example.</p>
</blockquote>

<blockquote>
  This is a blockquote with a source title.
  <small>Someone famous in <cite title = "Source Title">Source Title</cite></small>
</blockquote>

<blockquote class = "pull-right">This is a blockquote aligned to the right.
  <small>Someone famous in <cite title = "Source Title">Source Title</cite></small>
</blockquote>
```

Buttons

Class	Description
btn	Default/ Standard button.
btn-primary	Provides extra visual weight and identifies the primary action in a set of buttons.
btn-success	Indicates a successful or positive action.
btn-info	Contextual button for informational alert messages.
btn-warning	Indicates caution should be taken with this action.
btn-danger	Indicates a dangerous or potentially negative action.
btn-link	Deemphasize a button by making it look like a link while maintaining button behavior.

```
<!-- Standard button -->
<button type = "button" class = "btn btn-default">Default Button</button>

<!-- Provides extra visual weight and identifies the primary action in a set of buttons -->
<button type = "button" class = "btn btn-primary">Primary Button</button>

<!-- Indicates a successful or positive action -->
<button type = "button" class = "btn btn-success">Success Button</button>

<!-- Contextual button for informational alert messages -->
<button type = "button" class = "btn btn-info">Info Button</button>

<!-- Indicates caution should be taken with this action -->
<button type = "button" class = "btn btn-warning">Warning Button</button>

<!-- Indicates a dangerous or potentially negative action -->
<button type = "button" class = "btn btn-danger">Danger Button</button>

<!-- Deemphasize a button by making it look like a link while maintaining button behavior -->
<button type = "button" class = "btn btn-link">Link Button</button>
```

Button Size

The following table summarizes the classes used to get buttons of various sizes –

Class	Description
.btn-lg	This makes the button size large.
.btn-sm	This makes the button size small.

.btn-xs	This makes the button size extra small.
.btn-block	This creates block level buttons—those that span the full width of a parent.

The following example demonstrates this –

```

<p>
  <button type = "button" class = "btn btn-primary btn-lg">
    Large Primary button
  </button>

  <button type = "button" class = "btn btn-default btn-lg">
    Large button
  </button>
</p>

<p>
  <button type = "button" class = "btn btn-primary">
    Default size Primary button
  </button>

  <button type = "button" class = "btn btn-default">
    Default size button
  </button>
</p>

<p>
  <button type = "button" class = "btn btn-primary btn-sm">
    Small Primary 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 Primary button
  </button>

  <button type = "button" class = "btn btn-default btn-xs">
    Extra small button
  </button>
</p>

<p>
  <button type = "button" class = "btn btn-primary btn-lg btn-block">
    Block level Primary button
  </button>

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

```

Button State

Bootstrap provides classes which allow you to change the state of buttons as active, disabled etc. each of which are discussed in the following sections.

Active State

Buttons will appear pressed (with a darker background, darker border, and inset shadow) when active. The following table summarizes classes used to make button elements and anchor elements active –

Element	Class
Button element	Use .active class to show that it is activated.
Anchor element	Use .active class to <a> buttons to show that it is activated.

The following example demonstrates this –

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

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

<p>
  <button type = "button" class = "btn btn-primary btn-lg">
    Primary button
  </button>

  <button type = "button" class = "btn btn-primary btn-lg active">
    Active Primary button
  </button>
</p>
```

Disabled State

When you disable a button, it will fade in color by 50%, and lose the gradient.

The following table summarizes classes used to make button element and anchor element disabled –

Element	Class
---------	-------

Button element	Add the disabled <i>attribute</i> to <button> buttons.
Anchor element	Add the disabled <i>class</i> to <a> buttons. Note – This class will only change the <a>'s appearance, not its functionality. You need to use custom JavaScript to disable links here.

Responsive Images

Images comes in all sizes. So do screens. Responsive images automatically adjust to fit the size of the screen.

Create responsive images by adding an `.img-responsive` class to the `` tag. The image will then scale nicely to the parent element.

The `.img-responsive` class applies `display: block;` and `max-width: 100%;` and `height: auto;` to the image:

Forms

Bootstrap makes it easy with the simple HTML markup and extended classes for different styles of forms. In this chapter we will study how to create forms with ease using Bootstrap.

Form Layout

Bootstrap provides you with following types of form layouts –

1. Vertical (default) form
2. In-line form
3. Horizontal form

Vertical or Basic Form

The basic form structure comes with Bootstrap; individual form controls automatically receive some global styling. To create a basic form do the following –

- Add a role *form* to the parent `<form>` element.
- Wrap labels and controls in a `<div>` with class *.form-group*. This is needed for optimum spacing.
- Add a class of *.form-control* to all textual `<input>`, `<textarea>`, and `<select>` elements.

```
<form role = "form">
  <div class = "form-group">
    <label for = "name">Name</label>
```

```
<input type = "text" class = "form-control" id = "name" placeholder = "Enter Name">
</div>

<div class = "form-group">
  <label for = "inputfile">File input</label>
  <input type = "file" id = "inputfile">
  <p class = "help-block">Example block-level help text here.</p>
</div>

<div class = "checkbox">
  <label><input type = "checkbox"> Check me out</label>
</div>

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

Inline Form

To create a form where all of the elements are inline, left aligned and labels are alongside, add the class `.form-inline` to the `<form>` tag.

```
<form class = "form-inline" role = "form">

  <div class = "form-group">
    <label class = "sr-only" for = "name">Name</label>
    <input type = "text" class = "form-control" id = "name" placeholder = "Enter Name">
  </div>

  <div class = "form-group">
    <label class = "sr-only" for = "inputfile">File input</label>
    <input type = "file" id = "inputfile">
  </div>

  <div class = "checkbox">
    <label><input type = "checkbox"> Check me out</label>
  </div>

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

- By default inputs, selects, and textareas have 100% width in Bootstrap. You need to set a width on the form controls when using inline form.
- Using the class `.sr-only` you can hide the labels of the inline forms.

Horizontal Form

Horizontal forms stands apart from the others not only in the amount of markup, but also in the presentation of the form. To create a form that uses the horizontal layout, do the following –

- Add a class of `.form-horizontal` to the parent `<form>` element.
- Wrap labels and controls in a `<div>` with class `.form-group`.
- Add a class of `.control-label` to the labels.

```
<form class = "form-horizontal" role = "form">

  <div class = "form-group">
    <label for = "firstname" class = "col-sm-2 control-label">First Name</label>

    <div class = "col-sm-10">
      <input type = "text" class = "form-control" id = "firstname" placeholder = "Enter First Name">
    </div>
  </div>

  <div class = "form-group">
    <label for = "lastname" class = "col-sm-2 control-label">Last Name</label>
```

```

        <div class = "col-sm-10">
            <input type = "text" class = "form-control" id = "lastname" placeholder = "Enter
Last Name">
        </div>
    </div>

    <div class = "form-group">
        <div class = "col-sm-offset-2 col-sm-10">
            <div class = "checkbox">
                <label><input type = "checkbox"> Remember me</label>
            </div>
        </div>
    </div>

    <div class = "form-group">
        <div class = "col-sm-offset-2 col-sm-10">
            <button type = "submit" class = "btn btn-default">Sign in</button>
        </div>
    </div>

</form>

```

Supported Form Controls

Bootstrap natively supports the most common form controls mainly *input*, *textarea*, *checkbox*, *radio*, and *select*.

Inputs

The most common form text field is the input field. This is where users will enter most of the essential form data.

Bootstrap offers support for all native HTML5 input types: *text*, *password*, *datetime*, *datetime-local*, *date*, *month*, *time*, *week*, *number*, *email*, *url*, *search*, *tel*, and *color*. Proper *type* declaration is required to make *Inputs* fully styled.

```

<form role = "form">

    <div class = "form-group">
        <label for = "name">Label</label>
        <input type = "text" class = "form-control" placeholder = "Text input">
    </div>

</form>

```

Textarea

The textarea is used when you need multiple lines of input. Change *rows* attribute as necessary (fewer rows = smaller box, more rows = bigger box).

```

<form role = "form">

    <div class = "form-group">
        <label for = "name">Text Area</label>
        <textarea class = "form-control" rows = "3"></textarea>
    </div>

</form>

```



```
</form>
```

CheckBoxes and Radio Buttons

Checkboxes and radio buttons are great when you want users to choose from a list of preset options.

- When building a form, use *checkbox* if you want the user to select any number of options from a list. Use *radio* if you want to limit the user to just one selection.
- Use *.checkbox-inline* or *.radio-inline* class to a series of checkboxes or radios for controls appear on the same line.

The following example demonstrates both (default and inline) types –

```
<label for = "name">Example of Default Checkbox and radio button </label>

<div class = "checkbox">
  <label>
    <input type = "checkbox" value = "">Option 1
  </label>
</div>

<div class = "checkbox">
  <label>
    <input type = "checkbox" value = "">Option 2
  </label>
</div>

<div class = "radio">
  <label>
    <input type = "radio" name = "optionsRadios" id = "optionsRadios1" value = "option1"
checked> Option 1
  </label>
</div>

<div class = "radio">
  <label>
    <input type = "radio" name = "optionsRadios" id = "optionsRadios2" value = "option2">
Option 2 - selecting it will deselect option 1
  </label>
</div>

<label for = "name">Example of Inline Checkbox and radio button </label>

<div>
  <label class = "checkbox-inline">
    <input type = "checkbox" id = "inlineCheckbox1" value = "option1"> Option 1
  </label>

  <label class = "checkbox-inline">
    <input type = "checkbox" id = "inlineCheckbox2" value = "option2"> Option 2
  </label>

  <label class = "checkbox-inline">
    <input type = "checkbox" id = "inlineCheckbox3" value = "option3"> Option 3
  </label>
</div>
```

```

</label>

<label class = "checkbox-inline">
  <input type = "radio" name = "optionsRadiosinline" id = "optionsRadios3" value = "option1"
checked> Option 1
</label>

<label class = "checkbox-inline">
  <input type = "radio" name = "optionsRadiosinline" id = "optionsRadios4" value = "option2">
Option 2
</label>
</div>

```

Selects

A select is used when you want to allow the user to pick from multiple options, but by default it only allows one.

- Use <select> for list options with which the user is familiar, such as states or numbers.
- Use *multiple* = "multiple" to allow the users to select more than one option.

The following example demonstrates both (select and multiple) types –

```

<form role = "form">

  <div class = "form-group">
    <label for = "name">Select list</label>

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

    <label for = "name">Mutiple Select list</label>

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

  </div>

</form>

```

Static Control

Use the class *.form-control-static* on a <p>, when you need to place plain text next to a form label within a horizontal form.

```

<form class = "form-horizontal" role = "form">
  <div class = "form-group">
    <label class = "col-sm-2 control-label">Email</label>

    <div class = "col-sm-10">
      <p class = "form-control-static">email@example.com</p>
    </div>

  </div>

  <div class = "form-group">
    <label for = "inputPassword" class = "col-sm-2 control-label">Password</label>

    <div class = "col-sm-10">
      <input type = "password" class = "form-control" id = "inputPassword" placeholder =
"Password">
    </div>

  </div>
</form>

```

Form Control States

In addition to the *:focus* (i.e., a user clicks into the input or tabs onto it) state, Bootstrap offers styling for disabled inputs and classes for form validation.

Input Focus

When an input receives *:focus*, the outline of the input is removed and a *box-shadow* is applied.

Disabled Inputs

If you need to disable an input, simply adding the *disabled* attribute will not only disable it; it will also change the styling and the mouse cursor when the cursor hovers over the element.

Disabled Fieldsets

Add the disabled attribute to a `<fieldset>` to disable all the controls within the `<fieldset>` at once.

Validation States

Bootstrap includes validation styles for errors, warnings, and success messages. To use, simply add the appropriate class (*.has-warning*, *.has-error*, or *.has-success*) to the parent element.

The following example demonstrates all the form control states –

```

<form class = "form-horizontal" role = "form">
  <div class = "form-group">

```

```

<label class = "col-sm-2 control-label">Focused</label>

<div class = "col-sm-10">
  <input class = "form-control" id = "focusedInput" type = "text" value = "This is
focused...">
</div>
</div>

<div class = "form-group">
  <label for = "inputPassword" class = "col-sm-2 control-label">Disabled</label>

  <div class = "col-sm-10">
    <input class = "form-control" id = "disabledInput" type = "text" placeholder = "Disabled
input here..." disabled>
  </div>
</div>

<fieldset disabled>
  <div class = "form-group">

    <label for = "disabledTextInput" class = "col-sm-2 control-label">
      Disabled input (Fieldset disabled)
    </label>

    <div class = "col-sm-10">
      <input type = "text" id = "disabledTextInput" class = "form-control" placeholder =
"Disabled input">
    </div>
  </div>

  <div class = "form-group">
    <label for = "disabledSelect" class = "col-sm-2 control-label">
      Disabled select menu (Fieldset disabled)
    </label>

    <div class = "col-sm-10">
      <select id = "disabledSelect" class = "form-control">
        <option>Disabled select</option>
      </select>
    </div>

  </div>
</fieldset>

<div class = "form-group has-success">
  <label class = "col-sm-2 control-label" for = "inputSuccess">
    Input with success
  </label>

  <div class = "col-sm-10">
    <input type = "text" class = "form-control" id = "inputSuccess">
  </div>
</div>

<div class = "form-group has-warning">
  <label class = "col-sm-2 control-label" for = "inputWarning">
    Input with warning
  </label>

```

```

    <div class = "col-sm-10">
      <input type = "text" class = "form-control" id = "inputWarning">
    </div>
  </div>

  <div class = "form-group has-error">
    <label class = "col-sm-2 control-label" for = "inputError">
      Input with error
    </label>

    <div class = "col-sm-10">
      <input type = "text" class = "form-control" id = "inputError">
    </div>

  </div>
</form>

```

Form Control Sizing

You can set heights and widths of forms using classes like *.input-lg* and *.col-lg-** respectively. The following example demonstrates this –

```

<form role = "form">

  <div class = "form-group">
    <input class = "form-control input-lg" type = "text" placeholder = ".input-lg">
  </div>

  <div class = "form-group">
    <input class = "form-control" type = "text" placeholder = "Default input">
  </div>

  <div class = "form-group">
    <input class = "form-control input-sm" type = "text" placeholder = ".input-sm">
  </div>

  <div class = "form-group"></div>

  <div class = "form-group">
    <select class = "form-control input-lg">
      <option value = "">.input-lg</option>
    </select>
  </div>

  <div class = "form-group">
    <select class = "form-control">
      <option value = "">Default select</option>
    </select>
  </div>

  <div class = "form-group">
    <select class = "form-control input-sm">
      <option value = "">.input-sm</option>
    </select>
  </div>

```

```

</div>

<div class = "row">
  <div class = "col-lg-2">
    <input type = "text" class = "form-control" placeholder = ".col-lg-2">
  </div>

  <div class = "col-lg-3">
    <input type = "text" class = "form-control" placeholder = ".col-lg-3">
  </div>

  <div class = "col-lg-4">
    <input type = "text" class = "form-control" placeholder = ".col-lg-4">
  </div>

</div>
</form>

```

Help Text

Bootstrap form controls can have a block level help text that flows with the inputs. To add a full width block of content, use the *.help-block* after the `<input>`. The following example demonstrates this –

```

<form role = "form">
  <span>Example of Help Text</span>
  <input class = "form-control" type = "text" placeholder = "">
  <span class = "help-block">
    A longer block of help text that breaks onto a new line and may extend beyond one line.
  </span>
</form>

```

Close icon

Use the generic close icon for dismissing content like modals and alerts. Use the class **close** to get the close icon.

```

<p>Close Icon Example
  <button type = "button" class = "close" aria-hidden = "true">
    &times;
  </button>
</p>

```

Carets

Use carets to indicate dropdown functionality and direction. To get this functionality use the class **caret** with a `` element.

```
<p>Caret Example<span class = "caret"></span></p>
```

Quick Floats

You can float an element to the left or right with class **pull-left** or **pull-right** respectively the following example demonstrates this.

```
<div class = "pull-left">Quick Float to left</div>
<div class = "pull-right">Quick Float to right</div>
```

To align components in navbars with utility classes, use **.navbar-left** or **.navbar-right** instead. See the navbar chapter for details.

Center Content Blocks

Use class **center-block** to set an element to center.

```
<div class = "row">
  <div class = "center-block" style = "width:200px; background-color:#ccc;">
    This is an example for center-block
  </div>
</div>
```

Clearfix

To clear the float of any element, use the **.clearfix** class.

```
<div class = "clearfix" style = "background: #D8D8D8;border: 1px solid #000; padding: 10px;">

  <div class = "pull-left" style = "background:#58D3F7;">
    Quick Float to left
  </div>

  <div class = "pull-right" style = "background: #DA81F5;">
    Quick Float to right
  </div>

</div>
```

Showing and Hiding Content

You can force an element to be shown or hidden (including for screen readers) with the use of classes **.show** and **.hidden**.

```
<div class = "row" style = "padding: 91px 100px 19px 50px;">

  <div class = "show" style = "left-margin:10px; width:300px; background-color:#ccc;">
    This is an example for show class
  </div>

  <div class = "hidden" style = "width:200px; background-color:#ccc;">
    This is an example for hide class
  </div>

</div>
```

Screen Reader Content

You can hide an element to all devices except screen readers with the class **.sr-only**.

```
<div class = "row" style = "padding: 91px 100px 19px 50px;">
  <form class = "form-inline" role = "form">

    <div class = "form-group">
      <label class = "sr-only" for = "email">Email address</label>
      <input type = "email" class = "form-control" placeholder = "Enter email">
    </div>

    <div class = "form-group">
      <label class = "sr-only" for = "pass">Password</label>
      <input type = "password" class = "form-control" placeholder = "Password">
    </div>

  </form>
</div>
```

Glyphicons

What are Glyphicons?

Glyphicons are icon fonts which you can use in your web projects. [Glyphicons Halflings](#) are not free and require licensing, however their creator has made them available for Bootstrap projects free of cost.

"It is recommended, as a thank you, we ask you to include an optional link back to GLYPHICONS whenever practical". — Bootstrap Documentation

Where to find Glyphicons?

Now that we have downloaded Bootstrap 3.x version and understand its directory structure from the chapter Environment Setup, glyphicons can be found within the *fonts* folder. This contains the following files –

- glyphicons-halflings-regular.eot

- `glyphicons-halflings-regular.svg`
- `glyphicons-halflings-regular.ttf`
- `glyphicons-halflings-regular.woff`

Associated CSS rules are present within *bootstrap.css* and *bootstrap-min.css* files within `css` folder of *dist* folder. You can see the available glyphs at this link [GLYPHICONS](#).

Usage

To use the icons, simply use the following code just about anywhere in your code. Leave a space between the icon and text for proper padding.

```
<span class = "glyphicon glyphicon-search"></span>
```

The following example demonstrates this –

```
<p>
  <button type = "button" class = "btn btn-default">
    <span class = "glyphicon glyphicon-sort-by-attributes"></span>
  </button>

  <button type = "button" class = "btn btn-default">
    <span class = "glyphicon glyphicon-sort-by-attributes-alt"></span>
  </button>

  <button type = "button" class = "btn btn-default">
    <span class = "glyphicon glyphicon-sort-by-order"></span>
  </button>

  <button type = "button" class = "btn btn-default">
    <span class = "glyphicon glyphicon-sort-by-order-alt"></span>
  </button>
</p>

<button type = "button" class = "btn btn-default btn-lg">
  <span class = "glyphicon glyphicon-user"></span>

  User
</button>

<button type = "button" class = "btn btn-default btn-sm">
  <span class = "glyphicon glyphicon-user"></span>

  User
</button>

<button type = "button" class = "btn btn-default btn-xs">
  <span class = "glyphicon glyphicon-user"></span>

  User
</button>
```

Example

```
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Example of Bootstrap 3 Prepended and Appended Inputs</title>
<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css">
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap-
theme.min.css">
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js"></script>
<style type="text/css">
    .bs-example{
        margin: 20px;
    }
</style>
</head>
<body>
<div class="bs-example">
    <form>
        <div class="row">
            <div class="col-xs-4">
                <div class="input-group">
                    <span class="input-group-addon"><span class="glyphicon glyphicon-
user"></span></span>
                    <input type="text" class="form-control" placeholder="Username">
                </div>
            </div>
            <div class="col-xs-4">
                <div class="input-group">
                    <input type="text" class="form-control" placeholder="Amount">
                    <span class="input-group-addon">.00</span>
                </div>
            </div>
            <div class="col-xs-4">
                <div class="input-group">
                    <span class="input-group-addon">$</span>
                    <input type="text" class="form-control" placeholder="US Dollar">
                    <span class="input-group-addon">.00</span>
                </div>
            </div>
        </div>
    </form>
    <hr>
    <form>
        <div class="input-group">
            <span class="input-group-addon"><span class="glyphicon glyphicon-
user"></span></span>
            <input type="text" class="form-control" placeholder="Username">
        </div>
        <br>
        <div class="input-group">
            <input type="text" class="form-control" placeholder="Amount">
            <span class="input-group-addon">.00</span>
        </div>
    </form>
</div>
```

```

        </div>
        <br>
        <div class="input-group">
            <span class="input-group-addon">$</span>
            <input type="text" class="form-control" placeholder="US Dollar">
            <span class="input-group-addon">.<span>00</span></span>
        </div>
    </form>
</div>
</body>
</html>

```

Navbar Component

Creating a Simple Navbar with Bootstrap

You can use the Bootstrap navbar component to create responsive navigation header for your website or application. These responsive navbar initially collapsed on devices having small viewports like cell-phones but expand when user click the toggle button. However, it will be horizontal as normal on the medium and large devices like laptop or desktop.

You can also create different variations of the navbar such as navbars with dropdown menus and search boxes as well as fixed positioned navbar with much less effort. The following example will show you how to create a simple static navbar with navigation links.

```

<nav role="navigation" class="navbar navbar-default">
    <!-- Brand and toggle get grouped for better mobile display -->
    <div class="navbar-header">
        <button type="button" data-target="#navbarCollapse" data-toggle="collapse" class="navbar-toggle">
            <span class="sr-only">Toggle navigation</span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
        </button>
        <a href="#" class="navbar-brand">Brand</a>
    </div>
    <!-- Collection of nav links and other content for toggling -->
    <div id="navbarCollapse" class="collapse navbar-collapse">
        <ul class="nav navbar-nav">
            <li class="active"><a href="#">Home</a></li>
            <li><a href="#">Profile</a></li>
            <li><a href="#">Messages</a></li>
        </ul>
        <ul class="nav navbar-nav navbar-right">
            <li><a href="#">Login</a></li>
        </ul>
    </div>
</nav>

```

Note: Use the classes `.navbar-left` or `.navbar-right` instead of `.pull-left` or `.pull-right` to align the nav links, forms, buttons or text inside the navbar

Navbar with Dropdown and Search

```
<nav role="navigation" class="navbar navbar-default">
  <!-- Brand and toggle get grouped for better mobile display -->
  <div class="navbar-header">
    <button type="button" data-target="#navbarCollapse" data-toggle="collapse"
      class="navbar-toggle">
      <span class="sr-only">Toggle navigation</span>
      <span class="icon-bar"></span>
      <span class="icon-bar"></span>
      <span class="icon-bar"></span>
    </button>
    <a href="#" class="navbar-brand">Brand</a>
  </div>
  <!-- Collection of nav links, forms, and other content for toggling -->
  <div id="navbarCollapse" class="collapse navbar-collapse">
    <ul class="nav navbar-nav">
      <li class="active"><a href="#">Home</a></li>
      <li><a href="#">Profile</a></li>
      <li class="dropdown">
        <a data-toggle="dropdown" class="dropdown-toggle" href="#">Messages <b
          class="caret"></b></a>
        <ul role="menu" class="dropdown-menu">
          <li><a href="#">Inbox</a></li>
          <li><a href="#">Drafts</a></li>
          <li><a href="#">Sent Items</a></li>
          <li class="divider"></li>
          <li><a href="#">Trash</a></li>
        </ul>
      </li>
    </ul>
    <form role="search" class="navbar-form navbar-left">
      <div class="form-group">
        <input type="text" placeholder="Search" class="form-control">
      </div>
    </form>
    <ul class="nav navbar-nav navbar-right">
      <li><a href="#">Login</a></li>
    </ul>
  </div>
</nav>
```

Bootstrap Fixed Navbar

Bootstrap also provides mechanism to create navbar that is fixed on the top or bottom of the viewport and will scroll with the content on the page.

Creating Navbar Fixed to Top

Add an extra class `.navbar-fixed-top` in addition to the `.navbar` and `.navbar-default` base class to create navbars that is fixed on the top.

```
<nav role="navigation" class="navbar navbar-default navbar-fixed-top">
  <div class="container">
    <!-- Brand and toggle get grouped for better mobile display -->
    <div class="navbar-header">
      <button type="button" data-target="#navbarCollapse" data-toggle="collapse"
        class="navbar-toggle">
        <span class="sr-only">Toggle navigation</span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </button>
      <a href="#" class="navbar-brand">Brand</a>
    </div>
    <!-- Collection of nav links and other content for toggling -->
    <div id="navbarCollapse" class="collapse navbar-collapse">
      <ul class="nav navbar-nav">
        <li class="active"><a href="#">Home</a></li>
        <li><a href="#">Profile</a></li>
        <li><a href="#">Messages</a></li>
      </ul>
      <ul class="nav navbar-nav navbar-right">
        <li><a href="#">Login</a></li>
      </ul>
    </div>
  </div>
</nav>
```

Tip: Place the fixed `.navbar` content inside the `.container` or `.container-fluid` for proper padding and alignment with the rest of the content.

Creating Navbar Fixed to Bottom

Similarly to create navbars that is fixed at the bottom add the class `.navbar-fixed-bottom`

```
<nav role="navigation" class="navbar navbar-default navbar-fixed-bottom">
  <div class="container-fluid">
    <!-- Brand and toggle get grouped for better mobile display -->
    <div class="navbar-header">
```

```

        <button type="button" data-target="#navbarCollapse" data-toggle="collapse"
            class="navbar-toggle">
            <span class="sr-only">Toggle navigation</span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
        </button>
        <a href="#" class="navbar-brand">Brand</a>
    </div>
    <!-- Collection of nav links and other content for toggling -->
    <div id="navbarCollapse" class="collapse navbar-collapse">
        <ul class="nav navbar-nav">
            <li class="active"><a href="#">Home</a></li>
            <li><a href="#">Profile</a></li>
            <li><a href="#">Messages</a></li>
        </ul>
        <ul class="nav navbar-nav navbar-right">
            <li><a href="#">Login</a></li>
        </ul>
    </div>
</div>
</nav>

```

Note: Remember to add padding (at least 70px) to the top or bottom of the <body>element to avoid the content to go underneath the navbar while implementing fixed top or bottom navbar. Also be sure to add your custom style sheet after the core Bootstrap CSS file, otherwise it may not work.

Bootstrap Static Top Navbar

You can also create full-width navbar that appears on the top but scrolls away with the page by adding the class .navbar-static-top. Unlike the .navbar-fixed-top class, you do not need to change any padding on the <body> element.

```

<nav role="navigation" class="navbar navbar-default navbar-static-top">
    <div class="container">
        <!-- Brand and toggle get grouped for better mobile display -->
        <div class="navbar-header">
            <button type="button" data-target="#navbarCollapse" data-toggle="collapse"
                class="navbar-toggle">
                <span class="sr-only">Toggle navigation</span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
            </button>
            <a href="#" class="navbar-brand">Brand</a>
        </div>
        <!-- Collection of nav links and other content for toggling -->
        <div id="navbarCollapse" class="collapse navbar-collapse">
            <ul class="nav navbar-nav">
                <li class="active"><a href="#">Home</a></li>
                <li><a href="#">Profile</a></li>
                <li><a href="#">Messages</a></li>
            </ul>

```

```

        </ul>
        <ul class="nav navbar-nav navbar-right">
            <li><a href="#">Login</a></li>
        </ul>
    </div>
</div>
</nav>

```

Bootstrap Navbar with Search Form

Search form is very common component of the navbars and you have seen it on various website quite often. Search form can be placed inside the navbar using the class `.navbar-form` on the `<form>` element.

```

<nav role="navigation" class="navbar navbar-default">
    <div class="navbar-header">
        <button type="button" data-target="#navbarCollapse" data-toggle="collapse"
            class="navbar-toggle">
            <span class="sr-only">Toggle navigation</span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
        </button>
        <a href="#" class="navbar-brand">Brand</a>
    </div>
    <div id="navbarCollapse" class="collapse navbar-collapse">
        <form role="search" class="navbar-form navbar-left">
            <div class="form-group">
                <input type="text" placeholder="Search" class="form-control">
            </div>
            <button type="submit" class="btn btn-default">Submit</button>
        </form>
    </div>
</nav>

```

Creating the Inverted Variation of a Navbar

You can also create inverted variation of the Bootstrap navbar by adding an extra class `.navbar-inverse` to the `.navbar` base class, without any further change in markup.

```

<nav role="navigation" class="navbar navbar-inverse">
    <!-- Brand and toggle get grouped for better mobile display -->
    <div class="navbar-header">
        <button type="button" data-target="#navbarCollapse" data-toggle="collapse"
            class="navbar-toggle">
            <span class="sr-only">Toggle navigation</span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
        </button>
        <a href="#" class="navbar-brand">Brand</a>
    </div>
    <div id="navbarCollapse" class="collapse navbar-collapse">
        <ul class="list-unstyled">
            <li><a href="#">Home</a></li>
            <li><a href="#">About</a></li>
            <li><a href="#">Contact</a></li>
        </ul>
    </div>
</nav>

```

```

</div>
<!-- Collection of nav links, forms, and other content for toggling -->
<div id="navbarCollapse" class="collapse navbar-collapse">
  <ul class="nav navbar-nav">
    <li class="active"><a href="#">Home</a></li>
    <li><a href="#">Profile</a></li>
    <li class="dropdown">
      <a data-toggle="dropdown" class="dropdown-toggle" href="#">Messages <b
        class="caret"></b></a>
      <ul role="menu" class="dropdown-menu">
        <li><a href="#">Inbox</a></li>
        <li><a href="#">Drafts</a></li>
        <li><a href="#">Sent Items</a></li>
        <li class="divider"></li>
        <li><a href="#">Trash</a></li>
      </ul>
    </li>
  </ul>
  <form role="search" class="navbar-form navbar-left">
    <div class="form-group">
      <input type="text" placeholder="Search" class="form-control">
    </div>
  </form>
  <ul class="nav navbar-nav navbar-right">
    <li><a href="#">Login</a></li>
  </ul>
</div>
</nav>

```

Nav tabs & Pills Component

Bootstrap provides an easy and quick way to create basic nav components like tabs and pills which are very flexible and elegant. All the Bootstrap's nav components—tabs and pills—share the same base markup and styles through the base `.nav` class.

Creating Basic Tabs with Bootstrap

The following example will show you how to create a basic tab component using Bootstrap.

```

<ul class="nav nav-tabs">
  <li class="active"><a href="#">Home</a></li>
  <li><a href="#">Profile</a></li>
  <li><a href="#">Messages</a></li>
</ul>

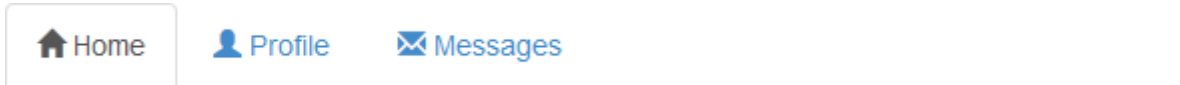
```



Adding Icons to Tabs

You can also add icons to your tabs to make it more attractive.

```
<ul class="nav nav-tabs">
  <li class="active"><a href="#"><span class="glyphicon glyphicon-home"></span>
    Home</a></li>
  <li><a href="#"><span class="glyphicon glyphicon-user"></span> Profile</a></li>
  <li><a href="#"><span class="glyphicon glyphicon-envelope"></span> Messages</a></li>
</ul>
```



Creating Basic Pills Nav with Bootstrap

You can create a basic pill based navigation using the base class `.nav-pills` instead of `.nav-tabs`, without any further change in markup.

```
<ul class="nav nav-pills">
  <li class="active"><a href="#">Home</a></li>
  <li><a href="#">Profile</a></li>
  <li><a href="#">Messages</a></li>
</ul>
```



Adding Icons to Pills Nav

You can also add icons to your pills nav to make it more attractive.

```
<ul class="nav nav-pills">
  <li class="active"><a href="#"><span class="glyphicon glyphicon-home"></span>
    Home</a></li>
  <li><a href="#"><span class="glyphicon glyphicon-user"></span> Profile</a></li>
  <li><a href="#"><span class="glyphicon glyphicon-envelope"></span>
    Messages</a></li>
</ul>
```



Stacked Pills Nav

Pills navigations are horizontal by default. To make them appear vertically stacked, just add an extra class `.nav-stacked` to the `` element.

```
<ul class="nav nav-pills nav-stacked">
  <li class="active"><a href="#">Home</a></li>
  <li><a href="#">Profile</a></li>
  <li><a href="#">Messages</a></li>
```


Home

Profile

Messages

Bootstrap Tabs and Pills Nav with Dropdown Menus

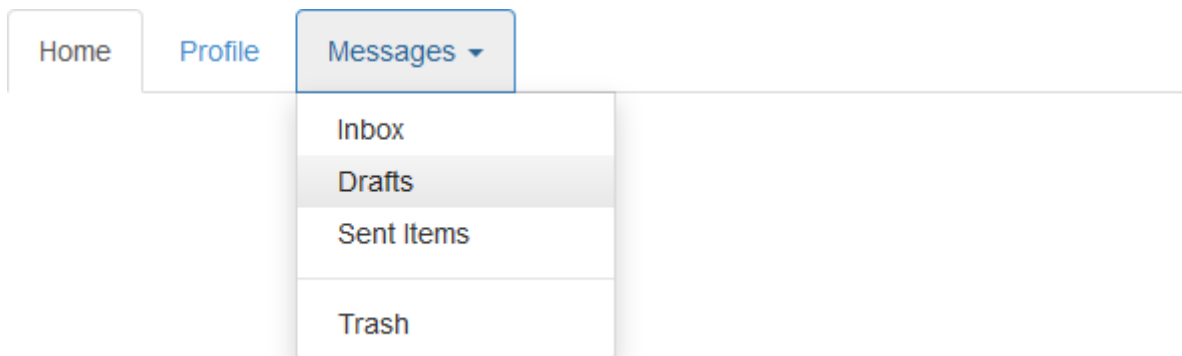
You can add dropdown menus to a link inside tabs and pills navigation with a little extra markup and including the JavaScript files `jquery.js` and `bootstrap.js` in your HTML file.

The four CSS classes `.dropdown`, `.dropdown-toggle`, `.dropdown-menu` and `.caret` are required in addition to the `.nav`, `.nav-tabs`, `.nav-pills` to create a simple dropdown menu.

Creating Tabs with Dropdowns

The following example will show you how to add simple dropdown menus to a tab

```
<ul class="nav nav-tabs">
  <li class="active"><a href="#">Home</a></li>
  <li><a href="#">Profile</a></li>
  <li class="dropdown">
    <a href="#" data-toggle="dropdown" class="dropdown-toggle">Messages <b
class="caret"></b></a>
    <ul class="dropdown-menu">
      <li><a href="#">Inbox</a></li>
      <li><a href="#">Drafts</a></li>
      <li><a href="#">Sent Items</a></li>
      <li class="divider"></li>
      <li><a href="#">Trash</a></li>
    </ul>
  </li>
</ul>
```



Creating Pills with Dropdowns

The following example will show you how to add simple dropdown menus to a pill nav.

```

<ul class="nav nav-pills">
  <li class="active"><a href="#">Home</a></li>
  <li><a href="#">Profile</a></li>
  <li class="dropdown">
    <a href="#" data-toggle="dropdown" class="dropdown-toggle">Messages <b
class="caret"></b></a>
    <ul class="dropdown-menu">
      <li><a href="#">Inbox</a></li>
      <li><a href="#">Drafts</a></li>
      <li><a href="#">Sent Items</a></li>
      <li class="divider"></li>
      <li><a href="#">Trash</a></li>
    </ul>
  </li>
</ul>

```

Breadcrumbs

Creating Breadcrumbs with Bootstrap

A breadcrumb is a navigation scheme that indicates current page's location to the user within a website or application. Breadcrumb navigation can greatly enhance the accessibility of the websites having a large number of pages.

```

<ul class="breadcrumb">
  <li><a href="#">Home</a></li>
  <li><a href="#">Products</a></li>
  <li class="active">Accessories</li>
</ul>

```

Home / Products / Accessories

Pagination

Pagination is the process of organizing content by dividing it into separate pages.

Pagination is used in some or other form quite often in almost every web application, for instance it is used by search engines for displaying a limited number of results on search results pages, or showing a limited number of posts for every page on a blog or forum.

```

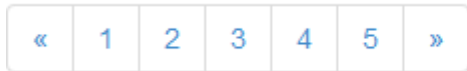
<ul class="pagination">
  <li><a href="#">&laquo;</a></li>

```

```

<li><a href="#">1</a></li>
<li><a href="#">2</a></li>
<li><a href="#">3</a></li>
<li><a href="#">4</a></li>
<li><a href="#">5</a></li>
<li><a href="#">&raquo;</a></li>
</ul>

```



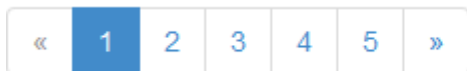
Pagination with Disabled and Active States

Links inside Bootstrap pagination can further be customized for different circumstances such as when user approaches to an end or start, or indicating current page number to the user. Use the class `.disabled` for making the links unclickable and `.active` to indicate the current page.

```

<ul class="pagination">
  <li class="disabled"><a href="#">&laquo;</a></li>
  <li class="active"><a href="#">1</a></li>
  <li><a href="#">2</a></li>
  <li><a href="#">3</a></li>
  <li><a href="#">4</a></li>
  <li><a href="#">5</a></li>
  <li><a href="#">&raquo;</a></li>
</ul>

```



Pager

Sometimes you may simply require previous and next links on your website to provide simple and quick navigation to the user instead of the complex pagination as we discussed above.

This can be done using the Bootstrap class `.pager`.

```
<ul class="pager">
  <li><a href="#">Previous</a></li>
  <li><a href="#">Next</a></li>
</ul>
```

[Previous](#)
[Next](#)

Creating Inline Labels

Inline labels are generally used to indicate some valuable information on the web pages such as important notes, warning messages, etc.

The following example will show you how to create inline labels using the Bootstrap.

```
<h1>Bootstrap heading <span class="label label-default">New</span></h1>
<h2>Bootstrap heading <span class="label label-default">New</span></h2>
<h3>Bootstrap heading <span class="label label-default">New</span></h3>
<h4>Bootstrap heading <span class="label label-default">New</span></h4>
<h5>Bootstrap heading <span class="label label-default">New</span></h5>
<h6>Bootstrap heading <span class="label label-default">New</span></h6>
```

Bootstrap heading New

Bootstrap heading New

Bootstrap heading New

Bootstrap heading New

Bootstrap heading New

Bootstrap heading New

There are some contextual classes to emphasize these inline labels.

```
<span class="label label-default">Default</span>
<span class="label label-primary">Primary</span>
<span class="label label-success">Success</span>
<span class="label label-info">Info</span>
<span class="label label-warning">Warning</span>
<span class="label label-danger">Danger</span>
```


Creating Inline Badges

Similarly you can create inline badges to provide important notification to the user such as number received or unread messages, number of friend requests etc. They're most commonly found in email client and social networking websites.

```
<ul class="nav nav-pills">
  <li><a href="#">Home</a></li>
  <li><a href="#">Profile</a></li>
  <li class="active"><a href="#">Messages <span class="badge">24</span></a></li>

  <li><a href="#">Notification <span class="badge">5</span></a></li>
</ul>
```



Using the Bootstrap Media Object

If you want to create a layout that contains left- or right-aligned media object like images or video alongside the textual content such as blog comments, Tweets, etc. you can do this easily using the newly introduced Bootstrap media component, like this:

```
<div class="media">
  <div class="media-left">
    <a href="#">
      
    </a>
  </div>
  <div class="media-body">
    <h4 class="media-heading">Jhon Carter <small><i>Posted on January 10,
    2014</i></small></h4>
    <p>Excellent feature! I love it. One day I'm definitely going to put this
    Bootstrap component into use and I'll let you know once I do.</p>
  </div>
</div>
```



Jhon Carter *Posted on January 10, 2014*

Excellent feature! I love it. One day I'm definitely going to put this Bootstrap component into use and I'll let you know once I do.

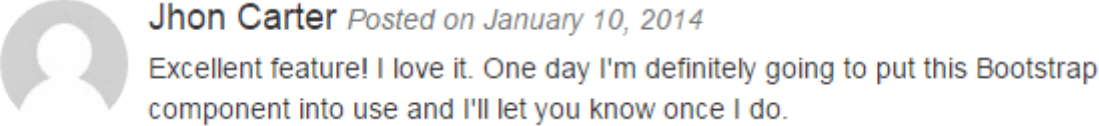
You can also apply image modifier classes like `.img-rounded`, `.img-circle` etc. to the embedded image to create other variation of the media component.

```
<div class="media">
  <div class="media-left">
    <a href="#">
      
    </a>
  </div>
  <div class="media-body">
    <h4 class="media-heading">Jhon Carter <small><i>Posted on January 10,
    2014</i></small></h4>
    <p>Excellent feature! I love it. One day I'm definitely going to put this
    Bootstrap component into use and I'll let you know once I do.</p>
  </div>
</div>
```

```

    </a>
  </div>
  <div class="media-body">
    <h4 class="media-heading">Jhon Carter <small><i>Posted on January 10,
    2014</i></small></h4>
    <p>Excellent feature! I love it. One day I'm definitely going to put
    this Bootstrap component into use and I'll let you know once I do.</p>
  </div>
</div>

```



Jumbotron

Showcasing Contents with Jumbotron

The Bootstrap jumbotron component provides an excellent way to showcase the key content or information on a web page. Just wrap your featured content like heading, descriptions etc. in a `<div>` element and apply the class `.jumbotron` on it.

```

<div class="jumbotron">
  <h1>Learn to Create Websites</h1>
  <p>In today's world internet is the most popular way...</p>
  <p><a href="#" class="btn btn-primary btn-lg">Learn more</a></p>
</div>

```

Creating Full Page Width Jumbotron

To create a jumbotron without rounded corners and that covers the full width of the viewport, place it outside all the containers and add the `.container` within like this.

```

<div class="jumbotron">
  <div class="container">
    <h1>Learn to Create Websites</h1>
    <p>In today's world internet is the most popular way...</p>
    <p><a href="#" class="btn btn-primary btn-lg">Learn more</a></p>
  </div>
</div>

```

Page header

A simple shell for an `h1` to appropriately space out and segment sections of content on a page. It can utilize the `h1`'s default `small` element, as well as most other components (with additional styles).

```
<div class="page-header">
  <h1>Example page header <small>Subtext for header</small></h1>
</div>
```

Thumbnails

Extend Bootstrap's grid system with the thumbnail component to easily display grids of images, videos, text, and more.

```
<div class="row">
  <div class="col-xs-6 col-md-3">
    <a href="#" class="thumbnail">
       </a>
    </div>
    ...
  </div>
```



Thumbnail Custom Example

```
<div class="row">
  <div class="col-sm-6 col-md-4">
    <div class="thumbnail">
      
      <div class="caption">
        <h3>Thumbnail label</h3>
        <p>...</p>
        <p><a href="#" class="btn btn-primary" role="button">Button</a>
          <a href="#" class="btn btn-default" role="button">Button</a>
        </p>
      </div>
    </div>
  </div>
</div>
```

Alerts

Provide contextual feedback messages for typical user actions with the handful of available and flexible alert messages.

```
<div class="alert alert-success" role="alert">...</div>
<div class="alert alert-info" role="alert">...</div>
<div class="alert alert-warning" role="alert">...</div>
<div class="alert alert-danger" role="alert">...</div>
```

Progress Bars

Creating Progress Bar with Bootstrap

Progress bars can be used for showing the progress of a task or action to the users. The following example will show you how to create a simple progress bar with vertical gradient.

```
<div class="progress">
  <div class="progress-bar" style="width: 60%;">
    <span class="sr-only">60% Complete</span>
  </div>
</div>
```



Creating Progress Bar with Label

To show to the progress status as a percentage label just remove the `` with `.sr-only` class from within the progress bar as demonstrated in example above.

Example

[Try this code »](#)

```
1. <div class="progress">
2.   <div class="progress-bar" style="width: 60%;">
3.     60%
4.   </div>
5. </div>
```

If you are showing percentage label you should also add a `min-width` to the progress bar to ensure that the label text remains readable even for low percentage, like this.

Example

[Try this code »](#)

```
• <div class="progress">
•   <div class="progress-bar" style="min-width: 20px;">
•     0%
•   </div>
• </div>
• <div class="progress">
•   <div class="progress-bar" style="min-width: 20px; width: 2%;">
```

- 2%
- `</div>`
- `</div>`

Creating Stripped Progress Bar

To create the stripped progress bar just add an extra class `.progress-striped` to the `.progress` base class.

Example

[Try this code »](#)

1. `<div class="progress progress-striped">`
2. `<div class="progress-bar" style="width: 60%;">`
3. `60% Complete`
4. `</div>`
5. `</div>`

— The output of the above example will look something like this:



Warning: The stripped progress bar uses a gradient to create the striped effect. The stripped progress bar is not supported in IE7-8.

Similarly you can create the animated progress bar — just add the `.active` class to `.progress-striped`. The `.active` class animates the stripes from right to left.

Creating Stacked Progress Bar

You can also place multiple bars into the same progress bar to stack them.

Example

[Try this code »](#)

- `<div class="progress">`
- `<div class="progress-bar progress-bar-success" style="width: 40%;">`
- `Program Files (40%)`
- `</div>`
- `<div class="progress-bar progress-bar-warning" style="width: 25%;">`
- `Residual Files (25%)`
- `</div>`
- `<div class="progress-bar progress-bar-danger" style="width: 15%;">`
- `Junk Files (15%)`
- `</div>`
- `</div>`

Progress Bars with Emphasis Classes

Bootstrap also provides some emphasis utility classes for progress bars that can be further used to convey meaning through color.

- `<div class="progress">`
- `<div class="progress-bar progress-bar-info" style="width: 20%;">`
- `20% Used`

- `</div>`
- `</div>`
- `<div class="progress">`
- `<div class="progress-bar progress-bar-success" style="width: 40%">`
- `40% Used`
- `</div>`
- `</div>`
- `<div class="progress">`
- `<div class="progress-bar progress-bar-warning" style="width: 80%">`
- `80% Used`
- `</div>`
- `</div>`
- `<div class="progress">`
- `<div class="progress-bar progress-bar-danger" style="width: 90%">`
- `90% Used`
- `</div>`
- `</div>`

List Groups

Creating List Groups with Bootstrap

The list groups are very useful and flexible component for displaying lists of elements in a beautiful manner. In most basic form a list group is simply an [unordered list](#) with the class `.list-group` whereas, the list items having the class `.list-group-item`.

```
<ul class="list-group">
  <li class="list-group-item">Pictures</li>
  <li class="list-group-item">Documents</li>
  <li class="list-group-item">Music</li>
  <li class="list-group-item">Videos</li>
</ul>
```

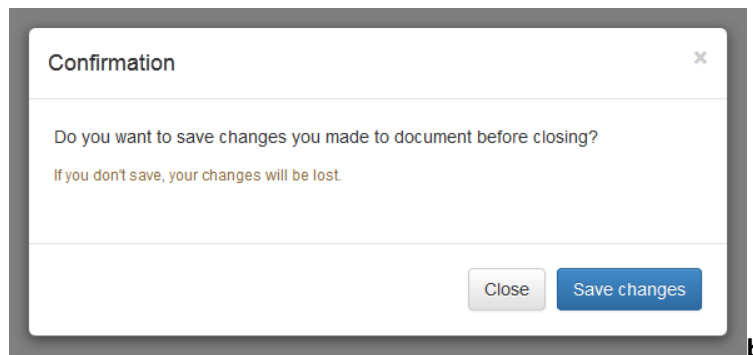
Bootstrap Modals

Creating Modals with Bootstrap

Modals are basically a dialog box that is used to provide important information to the user or prompt user to take necessary actions before moving on. Modal windows are widely used to warn users for situations like session time out or to receive their final confirmation before going to perform any critical actions such as saving or deleting important data.

You can easily create very smart and flexible dialog boxes with the Bootstrap modal plugin. The following example will show you how to create a simple modal with a header, message body and the footer containing action buttons for the user.

```
<div id="myModal" class="modal fade">
  <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">Confirmation</h4>
      </div>
      <div class="modal-body">
        <p>Do you want to save changes you made to document before closing?</p>
        <p class="text-warning"><small>If you don't save, your changes will be
          lost.</small></p>
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-default"
          data-dismiss="modal">Close</button>
        <button type="button" class="btn btn-primary">Save changes</button>
      </div>
    </div>
  </div>
</div>
```



Activate Modals via Data Attributes

You can activate a Bootstrap modal by clicking on the button or link via data attributes without writing any JavaScript code. See the following example:

```
<!-- Button HTML (to Trigger Modal) -->
<a href="#myModal" role="button" class="btn btn-large btn-primary" data-
toggle="modal">Launch Demo Modal</a>

<!-- Modal HTML -->
<div id="myModal" class="modal fade">
  <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">Confirmation</h4>
  </div>
  <div class="modal-body">
    <p>Do you want to save changes you made to document before closing?</p>
    <p class="text-warning"><small>If you don't save, your changes will be
      lost.</small></p>
  </div>
  <div class="modal-footer">
    <button type="button" class="btn btn-default" data-
      dismiss="modal">Close</button>
    <button type="button" class="btn btn-primary">Save changes</button>
  </div>
</div>
</div>
</div>

```

The above example launches the modal window on click of the "Launch Demo Modal" button. Let's go through each part of this modal code one by one for a better understanding.

Explanation of Code

To activate a Bootstrap modal via data attributes we basically need two components — the controller element like a button or link, and the modal element itself.

The outermost container of every modal in a document must have a unique id (in this case id="myModal"), so that it can be targeted via data-target (for buttons) or href(for hyperlinks) attribute of the controller element.

The attribute data-toggle="modal" is required to add on the controller element, like a button or an anchor, along with a attribute data-target="#myModal" or href="#myModal" to target a specific modal to toggle.

The .modal-dialog class sets the width as well as horizontal and vertical alignment of the modal box. Whereas the class .modal-content sets the styles like text and background color, borders, rounded corners etc.

Rest of the thing is self-explanatory, such as the .modal-header element defines a header for the modal that usually contains a modal title and a close button, whereas the .modal-body element contains the actual content like text, images, forms etc. and the .modal-footer element defines the footer that typically contains action buttons for the user.

Note: The .fade class on the .modal element adds a fading and sliding animation effect while showing and hiding the modal window. If you want the modal that simply appear without any effect you can just remove this class.

Activate Modals via JavaScript

You may also activate a Bootstrap modal window via JavaScript — just call the modal() Bootstrap method with the modal "id" or "class" selector in your JavaScript code

```

<script type="text/javascript">

```

```
$(document).ready(function(){
    $(".btn").click(function(){
        $("#myModal").modal('show');
    });
});
</script>
```

Changing the Sizes of Modals

Bootstrap gives you option further to scaling a modal up or down. You can make modals larger or smaller by adding an extra class `.modal-lg` or `.modal-sm` on `.modal-dialog`.

```
<!-- Large modal -->
<button class="btn btn-primary" data-toggle="modal" data-target="#largeModal">Large
modal</button>

<div id="largeModal" class="modal fade bs-example-modal-lg" tabindex="-1" role="dialog">
    <div class="modal-dialog modal-lg">
        <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">Large Modal</h4>
            </div>
            <div class="modal-body">
                <p>Add the <code>.modal-lg</code> class on <code>.modal-dialog</code> to
create this large modal.</p>
            </div>
            <div class="modal-footer">
                <button type="button" class="btn btn-default" data-
dismiss="modal">Cancel</button>
                <button type="button" class="btn btn-primary">OK</button>
            </div>
        </div>
    </div>
</div>

<!-- Small modal -->
<button class="btn btn-primary" data-toggle="modal" data-target="#smallModal">Small
modal</button>

<div id="smallModal" class="modal fade" tabindex="-1" role="dialog">
    <div class="modal-dialog modal-sm">
        <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">Small Modal</h4>
            </div>
            <div class="modal-body">
```

```

        <p>Add the <code>.modal-sm</code> class on <code>.modal-dialog</code> to
        create this small modal.</p>
    </div>
    <div class="modal-footer">
        <button type="button" class="btn btn-default" data-
dismiss="modal">Cancel</button>
        <button type="button" class="btn btn-primary">OK</button>
    </div>
</div>
</div>
</div>

```

Changing Modal Content Based on Trigger Button

Often several modal on a web page has almost same content with minor differences.

You can use the modal events to create slightly different modal windows based on the same modal HTML. The following example will change the title of the modal window according to the trigger button's data-title attribute value.

```

<script type="text/javascript">
$(document).ready(function(){
    $("#myModal").on('show.bs.modal', function(event){
        // Get button that triggered the modal
        var button = $(event.relatedTarget);
        // Extract value from data-* attributes
        var titleData = button.data('title');
        $(this).find('.modal-title').text(titleData + ' Form');
    });
});
</script>

```

Complete Example

```

<!DOCTYPE html>
<html lang="en">
<head>
    //..add necessary links for .js and .css files
$(document).ready(function(){
    $("#myModal").on('show.bs.modal', function(event){
        var button = $(event.relatedTarget); // Button that triggered the modal
        var titleData = button.data('title'); // Extract value from data-* attributes
        $(this).find('.modal-title').text(titleData + ' Form');
    });
});

```



```

});
</script>
<style type="text/css">
    .bs-example{
        margin: 20px;
    }
</style>
</head>
<body>
<div class="bs-example">
    <!-- Button HTML (to Trigger Modal) -->
    <button type="button" class="btn btn-primary" data-toggle="modal" data-
target="#myModal" data-title="Feedback">Feedback</button>
    <button type="button" class="btn btn-primary" data-toggle="modal" data-
target="#myModal" data-title="Report Error">Report Error</button>
    <button type="button" class="btn btn-primary" data-toggle="modal" data-
target="#myModal" data-title="Contact Us">Contact Us</button>

    <!-- Modal HTML -->
    <div id="myModal" class="modal fade">
        <div class="modal-dialog">
            <div class="modal-content">
                <div class="modal-header">
                    <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">×</button>
                    <h4 class="modal-title">Modal Window</h4>
                </div>
                <div class="modal-body">
                    <form role="form">
                        <div class="form-group">
                            <label for="recipient-name" class="control-
label">Email:</label>
                            <input type="text" class="form-control" id="recipient-name">
                        </div>
                        <div class="form-group">
                            <label for="message-text" class="control-
label">Message:</label>
                            <textarea class="form-control" id="message-text"></textarea>
                        </div>
                    </form>

```

```

        </div>
        <div class="modal-footer">
            <button type="button" class="btn btn-default" data-
dismiss="modal">Cancel</button>
            <button type="button" class="btn btn-primary">Send</button>
        </div>
    </div>
</div>
</div>
</div>
</body>
</html>

```

Options

There are certain options which may be passed to `modal()` Bootstrap method to customize the functionality of a modal window.

Name	Type	Default Value	Description
Backdrop	boolean or the string 'static'	true	Includes a modal-backdrop (black overlay area) element. Alternatively, you may specify static for a backdrop which doesn't close the modal on click.
Keyboard	boolean	true	Closes the modal window on press of escape key.
Show	boolean	true	Shows the modal when initialized or activate.
remote	URL	false	Deprecated If a remote url is provided, content will be loaded one time via jQuery's load method and injected into the '.modal-content' div.

You may set these options either through the use of data attributes or JavaScript. For setting the modals options via data attributes, just append the option name to data-, like data-backdrop="static", data-keyboard="false" etc.

However, JavaScript is the more preferable way for setting these options as it prevents you from repetitive work. See the modal's method `.modal (options)` in the section below to know how to set the options for modals using the JavaScript.

If you're using the data api for setting the options for modal window, you may alternatively use the "href" attribute to provide the URL of remote source, like this:

```

<!-- Button HTML (to Trigger Modal) -->
<a href="remote.html" role="button" class="btn btn-large btn-primary" data-toggle="modal"
data-target="#myModal">Launch Demo Modal</a>

<!-- Modal HTML -->
<div id="myModal" class="modal fade">
    <div class="modal-dialog">
        <div class="modal-content">

```

```

        <!-- Content will be loaded here from "remote.php" file -->
    </div>
</div>
</div>

```

Methods

These are the standard bootstrap's modals methods:

.modal(options)

This method activates the content as a modal. It also allows you to set options for them.

The jQuery code in the following example will prevent the modal from closing when a user clicks on the backdrop i.e. black overlay area behind the modal.

```

<script type="text/javascript">
$(document).ready(function(){
    $(".launch-modal").click(function(){
        $("#myModal").modal({
            backdrop: 'static'
        });
    });
});
</script>

```

The following jQuery code will prevent the modal from closing on press of the escape key.

```

<script type="text/javascript">
$(document).ready(function(){
    $(".launch-modal").click(function(){
        $("#myModal").modal({
            keyboard: false
        });
    });
});
</script>

```

The jQuery code in the following example will create a modal in which content of the modal will be inserted from a remote file upon activation.

```

<script type="text/javascript">
$(document).ready(function(){
    $(".launch-modal").click(function(){
        $("#myModal").modal({
            remote: '../remote.php'
        });
    });
});
</script>

```

.modal('toggle')

This method toggles a modal window manually.

```
<script type="text/javascript">
$(document).ready(function(){
    $(".toggle-modal").click(function(){
        $("#myModal").modal('toggle');
    });
});
</script>
```

.modal('show')

This method can be used to open a modal window manually.

```
<script type="text/javascript">
$(document).ready(function(){
    $(".open-modal").click(function(){
        $("#myModal").modal('show');
    });
});
</script>
```

.modal('hide')

This method can be used to hide a modal window manually.

```
<script type="text/javascript">
$(document).ready(function(){
    $(".hide-modal").click(function(){
        $("#myModal").modal('hide');
    });
});
</script>
```

.modal('handleUpdate')

This method readjusts the modal's position to counter the jerk that is occurring due to the appearance of the viewport scrollbar in case if the modal height changes in such a way that it becomes higher than the viewport height while it is open.

A common example of this scenario is showing the hidden elements inside the modal via JavaScript or loading content inside the modal using Ajax after activation.

```
<script type="text/javascript">
$(document).ready(function(){
    $(".show-text").click(function(){
        $('#myModal').find(".lots-of-text").toggle();
        $('#myModal').modal('handleUpdate')
    });
});
</script>
```

```
});  
});  
</script>
```

Events

Bootstrap's modal class includes few events for hooking into modal functionality.

Event	Description
show.bs.modal	This event fires immediately when the show instance method is called.
shown.bs.modal	This event is fired when the modal has been made visible to the user. It will wait until the CSS transition process has been fully completed before getting fired.
hide.bs.modal	This event is fired immediately when the hide instance method has been called.
hidden.bs.modal	This event is fired when the modal has finished being hidden from the user. It will wait until the CSS transition process has been fully completed before getting fired.
loaded.bs.modal	This event is fired when the modal has loaded content using the <code>remote</code> option.

The following example displays an alert message to the user when fade out transition of the modal window has been fully completed.

```
<script type="text/javascript">  
$(document).ready(function(){  
    $("#myModal").on('hidden.bs.modal', function(){  
        alert("Modal window has been completely closed.");  
    });  
});  
</script>
```

How to align Bootstrap modal vertically center

By default, Bootstrap modal window is aligned to the top of page with some margin. But you can align it in the middle of the page vertically using a simple JavaScript trick as described in the example below. This solution will dynamically adjust the alignment of the modal and always keep it in the center of the page even if the user resizes the browser window.

```
<!DOCTYPE html>  
<html lang="en">  
<head>  
<title>Vertical Center Alignment of Bootstrap Modal Dialog</title>  
<link rel="stylesheet" href="css/bootstrap.min.css">  
<link rel="stylesheet" href="css/bootstrap-theme.min.css">  
<script src="https://code.jquery.com/jquery-1.11.2.min.js"></script>  
<script src="js/bootstrap.min.js"></script>  
<script type="text/javascript">  
$(document).ready(function(){  
    function alignModal(){
```

```

        var modalDialog = $(this).find(".modal-dialog");
        /* Applying the top margin on modal dialog to align it vertically center */
        modalDialog.css("margin-top", Math.max(0, ($(window).height() -
modalDialog.height()) / 2));
    }
    // Align modal when it is displayed
    $(".modal").on("shown.bs.modal", alignModal);

    // Align modal when user resize the window
    $(window).on("resize", function(){
        $(".modal:visible").each(alignModal);
    });
});
</script>
</head>
<body>
    <!-- Button HTML (to Trigger Modal) -->
    <p><a href="#myModal" class="btn btn-lg btn-primary" data-toggle="modal">Launch
Demo Modal</a></p>

    <!-- Modal HTML -->
    <div id="myModal" class="modal">
        <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">Confirmation</h4>
                </div>
                <div class="modal-body">
                    <p>Do you want to save changes you made to document before
closing?</p>
                    <p class="text-warning"><small>If you don't save, your changes
will be lost.</small></p>
                </div>
                <div class="modal-footer">
                    <button type="button" class="btn btn-default" data-
dismiss="modal">Close</button>
                    <button type="button" class="btn btn-primary">Save
changes</button>
                </div>
            </div>
        </div>
    </div>
</body>
</html>

```

The function `alignModal()` in the example above align the Bootstrap modal by simply applying the top margin on the `.modal-dialog` element when modal is shown to the user using the `shown.bs.modal` event. The value of the `margin-top` property is obtained by subtracting the height of modal dialog from the height of the browser window divided by 2.

Before applying the top margin we've passed the obtained value along with the `0` to `JavaScriptMath.max()` function that returns the largest number from given numbers. This is necessary because if a user resize the browser window in such a way that the height of the browser window becomes lower than the modal height then the result of the margin calculation becomes negative in such situation `Math.max()` function return `0` and prevents from applying the negative margin. We're also calling the `alignModal()` function every time when browser window is resized so that it is always be in center of the page.

How to change the default width of Bootstrap modal box

By default, Bootstrap modal are available in three different sizes — default, small and large. However, if you still want to customize the size of the modal window you can override the CSS width property of the `.modal-dialog` class to resize the default modal. Similarly, you can override the width property of `.modal-sm` and `.modal-lg` class to resize the small and large modal box respectively. Here's an example:

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Resizing the Bootstrap Modal</title>
<link rel="stylesheet" href="css/bootstrap.min.css">
<link rel="stylesheet" href="css/bootstrap-theme.min.css">
<script src="https://code.jquery.com/jquery-1.11.2.min.js"></script>
<script src="js/bootstrap.min.js"></script>
<style type="text/css">
    @media screen and (min-width: 768px) {
        .modal-dialog {
            width: 700px; /* New width for default modal */
        }
        .modal-sm {
            width: 350px; /* New width for small modal */
        }
    }
    @media screen and (min-width: 992px) {
        .modal-lg {
            width: 950px; /* New width for large modal */
        }
    }
</style>
</head>
<body>
<div class="bs-example">
    <!-- Button HTML (to Trigger Modal) -->
```

```

    <a href="#myModal" class="btn btn-lg btn-primary" data-toggle="modal">Launch Default
    Modal</a>

    <!-- Modal HTML -->
    <div id="myModal" class="modal">
        <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">Default Modal</h4>
                </div>
                <div class="modal-body">
                    <p>The default modal size has been changed. Now it is 100px wider than
                    previous 600px wide modal.</p>
                </div>
                <div class="modal-footer">
                    <button type="button" class="btn btn-primary" data-dismiss="modal">Ok,
                    I Understand</button>
                </div>
            </div>
        </div>
    </div>
</div>
</body>
</html>

```

ScrollSpy

The Bootstrap scrollspy is a navigation mechanism that automatically highlights the nav links based on the scroll position to indicate the visitor where they are currently on the page. The scrollspy will make your web page more elegant and accessible, if you are using the bookmark links for directing the visitors to the different sections of a page that has a huge amount of content. Here's a typical example of Bootstrap scrollspy.

```

<body data-spy="scroll" data-target="#myScrollspy">
<div class="container">
    <div class="row">
        <div class="col-sm-3" id="myScrollspy">
            <ul class="nav nav-tabs nav-stacked" data-offset-top="120" data-spy="affix">
                <li class="active"><a href="#section1">Section One</a></li>
                <li><a href="#section2">Section Two</a></li>
                <li><a href="#section3">Section Three</a></li>
            </ul>
        </div>
        <div class="col-sm-9">
            <div id="section1">
                <h2>Section One</h2>
            </div>
        </div>
    </div>
</div>

```



```

        <p>This is section one content...</p>
    </div>
    <hr>
    <div id="section2">
        <h2>Section Two</h2>
        <p>This is section two content...</p>
    </div>
    <hr>
    <div id="section3">
        <h2>Section Three</h2>
        <p>This is section three content...</p>
    </div>
</div>
</div>
</div>
</body>

```

Creating ScrollSpy via Data Attributes

You can easily add scrollspy behavior to your topbar navigation via data attributes without writing a single line of JavaScript code. Let's check out the following example:

```

<body data-spy="scroll" data-target="#myNavbar" data-offset="70">
    <nav id="myNavbar" class="navbar navbar-inverse navbar-fixed-top" role="navigation">
        <div class="container">
            <!-- Brand and toggle get grouped for better mobile display -->
            <div class="navbar-header">
                <button type="button" class="navbar-toggle" data-toggle="collapse" data-
                target="#navbarCollapse">
                    <span class="sr-only">Toggle navigation</span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                </button>
                <a class="navbar-brand" href="#">Scrollspy</a>
            </div>
            <!-- Collection of nav links, forms, and other content for toggling -->
            <div class="collapse navbar-collapse" id="navbarCollapse">
                <ul class="nav navbar-nav">
                    <li class="active"><a href="#section1">Section 1</a></li>
                    <li><a href="#section2">Section 2</a></li>
                    <li><a href="#section3">Section 3</a></li>
                    <li class="dropdown"><a href="#" class="dropdown-toggle" data-
                    toggle="dropdown">Section 4<b class="caret"></b></a>
                        <ul class="dropdown-menu">
                            <li><a href="#section4dot1">Section 4.1</a></li>

```

```

        <li><a href="#section4dot2">Section 4.2</a></li>
        <li><a href="#section4dot3">Section 4.3</a></li>
    </ul>
</li>
<li><a href="#section5">Section 5</a></li>
</ul>
</div>
</div>
</nav>
<div class="container">
    <div id="section1">
        <h2>Section 1</h2>
        <p>This is section 1 content...</p>
    </div>
    <hr>
    <div id="section2">
        <h2>Section 2</h2>
        <p>This is section 2 content...</p>
    </div>
    <hr>
    <div id="section3">
        <h2>Section 3</h2>
        <p>This is section 3 content...</p>
    </div>
    <hr>
    <h2>Section 4</h2>
    <p>This is section 4 content</p>
    <div id="section4dot1">
        <h3>Section 4.1</h3>
        <p>This is section 4.1 content...</p>
    </div>
    <div id="section4dot2">
        <h3>Section 4.2</h3>
        <p>This is section 4.2 content...</p>
    </div>
    <div id="section4dot3">
        <h3>Section 4.3</h3>
        <p>This is section 4.3 content...</p>
    </div>
    <hr>
    <div id="section5">
        <h2>Section 5</h2>
        <p>This is section 5 content...</p>
    </div>
</div>
</body>

```

Explanation of Code

The Bootstrap scrollspy has basically two components — the target nav (e.g. navbar, nav tabs or pills) and the scrollable area to spy on, which is often the `<body>` section.

- The `data-spy="scroll"` attribute is applied to the scrollable element that is being spied on, which is the `<body>` element.
- The `data-target` attribute is added on the scrollable element with the ID or class of the parent element of the Bootstrap `.nav` component so that nav links can be targeted by the scrollspy for highlighting purpose.
- The optional `data-offset` attribute specifies the number of pixels to offset from top when calculating the position of scroll. Adjust the offset value if the targeted links are highlighting too early or too late. The default value is 10 pixels.

Rest of the thing is self explanatory, such as the `.navbar` element specifies a [Bootstrap navbar](#), the element `<div id="section1"></div>` (line no-33) create a bookmark with the `id` attribute, whereas the element `Section 1` (line no-17) add a link to this bookmark, from within the same page, and so on.

Creating ScrollSpy via JavaScript

You may also add scrollspy manually using the JavaScript — just call the `scrollspy()` Bootstrap method with the "id" or "class" selector of the navbar in your JavaScript code.

```
<script type="text/javascript">
$(document).ready(function(){
    $("body").scrollspy({
        target: "#myNavbar",
        offset: 70
    })
});
</script>
```

Options

There are certain options which may be passed to `scrollspy()` Bootstrap method to customize the functionality of a scrollspy.

Name	Type	Default Value	Description
offset	number	10	Number of pixels to offset from top when calculating position of scroll.

You can also set this options for scrollspy using the data attributes — just append the option name to `data-`, like `data-offset="0"`.

Methods

These are the standard bootstrap's scrollspy methods:

.scrollspy('refresh')

When using scrollspy in conjunction with adding or removing of elements from the DOM, you'll need to call the refresh method like this:

```
<script type="text/javascript">
$(document).ready(function(){
    $('[data-spy="scroll"]').each(function(){
        var $spy = $(this).scrollspy('refresh');
    });
});
</script>
```

Events

Bootstrap's scrollspy class includes few events for hooking into scrollspy functionality.

Event	Description
activate.bs.scrollspy	This event fires whenever a new item becomes activated by the scrollspy.

The following example displays an alert message to the user when a new item becomes highlighted by the scrollspy.

```
<script type="text/javascript">
$(document).ready(function(){
    $("#myNavbar").on("activate.bs.scrollspy", function(){
        var currentItem = $(".nav li.active > a").text();
        $("#info").empty().html("Currently you are viewing - " + currentItem);
    });
});
</script>
```

Bootstrap Tabs

Tab based navigations provides an easy and powerful mechanism to handle huge amount of content within a small area through separating content into different panes where each pane is viewable one at a time.

The user can quickly access the content through switching between the panes without leaving the page. The following example will show you how to create the basic tabs using the Bootstrap tab component.

```
<ul class="nav nav-tabs">
  <li class="active"><a href="#">Home</a></li>
```

```

<li><a href="#">Profile</a></li>
<li><a href="#">Messages</a></li>
</ul>

```

Creating Dynamic Tabs via Data Attributes

You can activate a tab component without writing any JavaScript — simply specify the `data-toggle="tab"` on each tab, as well as create a `.tab-pane` with unique ID for every tab and wrap them in `.tab-content`.

```

<ul class="nav nav-tabs">
  <li class="active"><a data-toggle="tab" href="#sectionA">Section A</a></li>
  <li><a data-toggle="tab" href="#sectionB">Section B</a></li>
  <li class="dropdown">
    <a data-toggle="dropdown" class="dropdown-toggle" href="#">Dropdown <b
      class="caret"></b></a>
    <ul class="dropdown-menu">
      <li><a data-toggle="tab" href="#dropdown1">Dropdown 1</a></li>
      <li><a data-toggle="tab" href="#dropdown2">Dropdown 2</a></li>
    </ul>
  </li>
</ul>
<div class="tab-content">
  <div id="sectionA" class="tab-pane fade in active">
    <p>Section A content...</p>
  </div>
  <div id="sectionB" class="tab-pane fade">
    <p>Section B content...</p>
  </div>
  <div id="dropdown1" class="tab-pane fade">
    <p>Dropdown 1 content...</p>
  </div>
  <div id="dropdown2" class="tab-pane fade">
    <p>Dropdown 2 content...</p>
  </div>
</div>

```

Creating Dynamic Tabs via JavaScript

You may also enable tabs via JavaScript. Each tab needs to be activated individually

```

<script type="text/javascript">
$(document).ready(function(){
  $("#myTab a").click(function(e){
    e.preventDefault();
    $(this).tab('show');
  });
});

```

```
});  
</script>
```

Methods

This is the standard bootstrap's tab method:

`$.tab('show')`

Activates a tab element and the related content container. Tab should have either a data-target or an href for targeting a container node in the DOM.

```
<script type="text/javascript">  
$(document).ready(function(){  
    $("#myTab li:eq(1) a").tab('show');  
})  
</script>
```

Events

These are the standard Bootstrap events to enhance the tab functionality.

Event	Description
show.bs.tab	This event fires on tab show, but before the new tab has been shown. You can use the <code>event.target</code> and <code>event.relatedTarget</code> to target the active tab and the previous active tab (if available) respectively.
shown.bs.tab	This event fires on tab show after a tab has been shown. You can use the <code>event.target</code> and <code>event.relatedTarget</code> to target the active tab and the previous active tab (if available) respectively.
hide.bs.tab	This event fires when the current active tab is to be hidden and thus a new tab is to be shown. You can use the <code>event.target</code> and <code>event.relatedTarget</code> to target the current active tab and the new tab which is going to be active very soon, respectively.
hidden.bs.tab	This event fires after the previous active tab is hidden and a new tab is shown. You can use the <code>event.target</code> and <code>event.relatedTarget</code> to target the previous active tab and the new active tab, respectively.

The following example displays the names of active tab and previous tab to the user when transition of a tab has been fully completed.

```
<script type="text/javascript">  
$(document).ready(function(){  
    $('a[data-toggle="tab"]').on('shown.bs.tab', function(e){  
        var activeTab = $(e.target).text(); // Get the name of active tab  
        var previousTab = $(e.relatedTarget).text(); // Get the name of previous tab
```

```

$(".active-tab span").html(activeTab);
$(".previous-tab span").html(previousTab);
});
});
</script>

```

```

<ul class="nav nav-tabs" id="myTab">
  <li class="active"><a data-toggle="tab" href="#home">Home</a></li>
  <li><a data-toggle="tab" href="#profile">Profile</a></li>
  <li class="dropdown">
    <a data-toggle="dropdown" class="dropdown-toggle" href="#">Dropdown <b
class="caret"></b></a>
    <ul class="dropdown-menu">
      <li><a data-toggle="tab" href="#dropdown1">Dropdown1</a></li>
      <li><a data-toggle="tab" href="#dropdown2">Dropdown2</a></li>
    </ul>
  </li>
</ul>
<div class="tab-content" id="myTabContent">
  <div id="sectionA" class="tab-pane fade in active">
    <h3>Section A</h3>
    <p>Aliquip placeat salvia cillum iphone. Seitan aliquip quis cardigan
american apparel, butcher voluptate nisi qui. Raw denim you probably haven't heard of
them jean shorts Austin. Nesciunt tofu stumptown aliqua, retro synth master cleanse.
Mustache cliché tempor, williamsburg carles vegan helvetica. Reprehenderit butcher
retro keffiyeh dreamcatcher synth.</p>
  </div>
  <div id="sectionB" class="tab-pane fade">
    <h3>Section B</h3>
    <p>Vestibulum nec erat eu nulla rhoncus fringilla ut non neque. Vivamus
nibh urna, ornare id gravida ut, mollis a magna. Aliquam porttitor condimentum nisi,
eu viverra ipsum porta ut. Nam hendrerit bibendum turpis, sed molestie mi fermentum
id. Aenean volutpat velit sem. Sed consequat ante in rutrum convallis. Nunc facilisis
leo at faucibus adipiscing.</p>
  </div>
  <div id="dropdown1" class="tab-pane fade">
    <h3>Dropdown 1</h3>
    <p>WInteger convallis, nulla in sollicitudin placerat, ligula enim auctor
lectus, in mollis diam dolor at lorem. Sed bibendum nibh sit amet dictum feugiat.
Vivamus arcu sem, cursus a feugiat ut, iaculis at erat. Donec vehicula at ligula
vitae venenatis. Sed nunc nulla, vehicula non porttitor in, pharetra et dolor. Fusce
nec velit velit. Pellentesque consectetur eros.</p>
  </div>
  <div id="dropdown2" class="tab-pane fade">
    <h3>Dropdown 2</h3>
    <p>Donec vel placerat quam, ut euismod risus. Sed a mi suscipit,
elementum sem a, hendrerit velit. Donec at erat magna. Sed dignissim orci nec
eleifend egestas. Donec eget mi consequat massa vestibulum laoreet. Mauris et
ultrices nulla, malesuada volutpat ante. Fusce ut orci lorem. Donec molestie libero
in tempus imperdiet. Cum sociis natoque penatibus et magnis dis parturient.</p>

```

```

        </div>
    </div>
    <hr>
    <p class="active-tab"><strong>Active Tab</strong>: <span></span></p>
    <p class="previous-tab"><strong>Previous Tab</strong>: <span></span></p>

```

Bootstrap Tooltips

A tooltip is a small pop up that appears when user places the mouse pointer over an element such as link or buttons to provide hint or information about the element being hovered.

Tooltips can be very helpful for the new visitors of your website because they enable the user to know the purpose of icons and links by placing the mouse pointer over them.

Triggering the Tooltips

Tooltips can be triggered via JavaScript — just call the `tooltip()` Bootstrap method with the `id` or `class` selector of the target element in your JavaScript code.

```

<script type="text/javascript">
$(document).ready(function(){
    $('[data-toggle="tooltip"]').tooltip();
});
</script>

<ul class="list-inline">
    <li><a href="#" data-toggle="tooltip" title="Default tooltip">Tooltip</a></li>
    <li><a href="#" data-toggle="tooltip" title="Another tooltip">Another tooltip</a></li>
    <li><a href="#" data-toggle="tooltip" title="A much longer tooltip to demonstrate the max-
        width of the Bootstrap tooltip.">Large tooltip</a></li>
    <li><a href="#" data-toggle="tooltip" title="The last tip!">Last tooltip</a></li>
</ul>
</div>

```

Setting the Directions of Tooltips

You can set tooltips to appear on top, right, bottom and left sides of an element.

Positioning of Tooltips via Data Attributes

The following example will show you how to set the direction of tooltips via data attributes.

```

<a href="#" data-toggle="tooltip" data-placement="top" title="Default tooltip">Tooltip</a>
<a href="#" data-toggle="tooltip" data-placement="right" title="Another tooltip">Another tooltip</a>
<a href="#" data-toggle="tooltip" data-placement="bottom" title="A large tooltip.">Large tooltip</a>
<a href="#" data-toggle="tooltip" data-placement="left" title="The last tip!">Last tooltip</a>

```

Positioning of Tooltips via JavaScript

The following example will show you how to set the direction of tooltips via JavaScript.


```

<script type="text/javascript">
$(document).ready(function(){
    $(".tip-top").tooltip({placement : 'top'});
    $(".tip-right").tooltip({placement : 'right'});
    $(".tip-bottom").tooltip({placement : 'bottom'});
    $(".tip-left").tooltip({ placement : 'left'});
});
</script>

```

Options

There are certain options which may be passed to `tooltip()` Bootstrap method to customize the functionality of the tooltip plugin.

Name	Type	Default Value	Description
animation	boolean	True	Apply a CSS fade transition to the tooltip.
html	boolean	False	Insert html into the tooltip. If false, jQuery's <code>text()</code> method will be used to insert content into the DOM. Use <code>text</code> if you're worried about XSS attacks.
placement	string function	'top'	Sets the position of the tooltip — top bottom left right auto. When "auto" value is specified, it will dynamically reorient the tooltip. For example, if placement value is "auto top", the tooltip will display on the top when possible, otherwise it will display on the bottom.
selector	string	False	If a selector is provided, tooltip objects will be attached to the specified targets.
title	string function	"	Sets the default title value if title attribute isn't present.
trigger	string	'hover focus'	Specify how tooltip is triggered — click hover focus manual. Note you can pass trigger multiple, space separated, trigger types.
delay	number object	0	Time to delay in showing and hiding the tooltip (ms) — does not apply to manual trigger type. If a number is supplied, delay is applied to both hide/show Object structure is: delay: { show: 500, hide: 100 }
container	string false	false	Appends the tooltip to a specific element/container: 'body'
template	string	'<div class="tooltip" role="tooltip"><div class="tooltip-arrow"></div><div class="tooltip-inner"></div></div>'	Base HTML to use when creating the tooltip. The tooltip's title will be inserted into the element having the class <code>.tooltip-inner</code> and the element with the class <code>.tooltip-arrow</code> will become the tooltip's arrow. The outermost wrapper element should have the <code>.tooltip</code> class.

viewport	string object	{ selector: 'body', padding: 0 }	Keeps the tooltip within the bounds of this element. Example: viewport: '#viewport' or { selector: '#viewport', padding: 0 }
----------	-----------------	----------------------------------	---

You may set these options either through the use of data attributes or JavaScript. For setting the tooltips options via data attributes, just append the option name to data- along with the correct value, like data-animation="false", data-placement="bottom" etc.

However, JavaScript is the more preferable way for setting these options as it prevents you from repetitive work.

\$.tooltip(options)

This method attaches the tooltip handler to a group of elements. It also allows you to set the options for the tooltips, so that you can customize them according to your needs.

The following example will insert the specified text inside the tooltips if the value of the title attribute is omitted or missing from the selected elements:

```
<script type="text/javascript">
$(document).ready(function(){
    $("#myTooltips a").tooltip({
        title: 'It works in absence of title attribute.'
    });
});
</script>
```

The following example will show you how to place the HTML content inside a tooltip:

```
<script type="text/javascript">
$(document).ready(function(){
    $("#myTooltip").tooltip({
        title: "<h4><img src='images/smiley.png' alt='Smiley'> Hello, <b>I'm</b><br><i>Smiley!</i></h4>",
        html: true
    });
});
</script>
```

The following example will show you how to control the timing of showing and hiding of the tooltip using the tooltip's delay option via JavaScript.

```
<script type="text/javascript">
$(document).ready(function(){
    // Showing and hiding tooltip with same speed
    $(".tooltip-tiny").tooltip({
        delay: 100
    });

    // Showing and hiding tooltip with different speed
    $(".tooltip-large").tooltip({
        delay: {show: 0, hide: 500}
    });
});
</script>
```

The following example will show you how you can create your own custom template for the Bootstrap tooltips instead of using the default one.

```
<script type="text/javascript">
$(document).ready(function(){
    $("#myTooltips a").tooltip({
        template : '<div class="tooltip"><div class="tooltip-arrow"></div><div class="tooltip-head"><h3><span
class="glyphicon glyphicon-info-sign"></span> Tool Info</h3></div><div class="tooltip-inner"></div></div>'
    });
});
</script>

<style type="text/css">
    .bs-example{
        margin: 200px 100px;
    }
    .bs-example a{
        margin: 25px;
        font-size: 20px;
    }
    /* Styles for custom tooltip template */
    .tooltip-head{
        color: #fff;
        background: #000;
        padding: 10px 10px 5px;
        border-radius: 4px 4px 0 0;
        text-align: center;
        margin-bottom: -2px; /* Hide default tooltip rounded corner from top */
    }
    .tooltip-head .glyphicon{
        font-size: 22px;
        vertical-align: bottom;
    }
    .tooltip-head h3{
        margin: 0;
        font-size: 18px;
    }
</style>
</head>
<body>
<div class="bs-example">
    <div id="myTooltips">
        <a href="#" data-toggle="tooltip" title="Open this document for editing in a text editor."><span
class="glyphicon glyphicon-edit"></span></a>
    </div>
</div>
```

```

        <a href="#" data-toggle="tooltip" title="Save this document on the remote web server on a permanent
basis."><span class="glyphicon glyphicon-floppy-disk"></span></a>
        <a href="#" data-toggle="tooltip" title="Download this document from the remote server."><span
class="glyphicon glyphicon-download-alt"></span></a>
        <a href="#" data-toggle="tooltip" title="Print this document using a printer installed on your
machine."><span class="glyphicon glyphicon-print"></span></a>
        <a href="#" data-toggle="tooltip" title="Delete this document permanently."><span class="glyphicon
glyphicon-trash"></span></a>
        <a href="#" data-toggle="tooltip" title="Document Settings"><span class="glyphicon glyphicon-
cog"></span></a>
    </div>
</div>
</body>
</html>

```

.tooltip('show')

This method reveals an element's tooltip.

```

<script type="text/javascript">
$(document).ready(function(){
    $(".show-tooltip").click(function(){
        $("#myTooltip").tooltip('show');
    });
});
</script>

```

.tooltip('hide')

This method hides an element's tooltip.

```

<script type="text/javascript">
$(document).ready(function(){
    $(".hide-tooltip").click(function(){
        $("#myTooltip").tooltip('hide');
    });
});
</script>

```

.tooltip('toggle')

This method toggles an element's tooltip.

```

<script type="text/javascript">
$(document).ready(function(){
    $(".toggle-tooltip").click(function(){
        $("#myTooltip").tooltip('toggle');
    });
});
</script>

```

.tooltip('destroy')

This method hides and destroys an element's tooltip.

```
<script type="text/javascript">
$(document).ready(function(){
    $(".destroy-tooltip").click(function(){
        $("#myTooltip").tooltip('destroy');
    });
});
</script>
```

Events

Bootstrap's tooltip class includes few events for hooking into tooltip functionality.

Event	Description
show.bs.tooltip	This event fires immediately when the show instance method is called.
shown.bs.tooltip	This event is fired when the tooltip has been made visible to the user. It will wait until the CSS transition process has been fully completed before getting fired.
hide.bs.tooltip	This event is fired immediately when the hide instance method has been called.
hidden.bs.tooltip	This event is fired when the tooltip has finished being hidden from the user. It will wait until the CSS transition process has been fully completed before getting fired.
inserted.bs.tooltip	This event is fired after the show.bs.tooltip event when the tooltip template has been added to the DOM.

The following example will display an alert message to the user when the fade out transition of the tooltip has been fully completed.

```
<script type="text/javascript">
$(document).ready(function(){
    $('[data-toggle="tooltip"]').on('hidden.bs.tooltip', function(){
        alert("Tooltip has been completely closed.");
    });
});
</script>
```

Tables

Bootstrap provides a clean layout for building tables. Some of the table elements supported by Bootstrap are –

Tag	Description
<table>	Wrapping element for displaying data in a tabular format
<thead>	Container element for table header rows (<tr>) to label table columns.
<tbody>	Container element for table rows (<tr>) in the body of the table.
<tr>	Container element for a set of table cells (<td> or <th>) that appears on a single row.
<td>	Default table cell.

<th>	Special table cell for column (or row, depending on scope and placement) labels. Must be used within a <thead>
<caption>	Description or summary of what the table holds.

Basic Table

If you want a nice, basic table style with just some light padding and horizontal dividers, add the base class of `.table` to any table as shown in the following example –

```
<table class = "table">
  <caption>Basic Table Layout</caption>

  <thead>
    <tr>
      <th>Name</th>
      <th>City</th>
    </tr>
  </thead>

  <tbody>
    <tr>
      <td>Tanmay</td>
      <td>Bangalore</td>
    </tr>

    <tr>
      <td>Sachin</td>
      <td>Mumbai</td>
    </tr>
  </tbody>
</table>
```

Optional Table Classes

Along with the base table markup and the `.table` class, there are a few additional classes that you can use to style the markup. Following sections will give you a glimpse of all these classes.

Striped Table

By adding the `.table-striped` class, you will get stripes on rows within the `<tbody>`

Bordered Table

By adding the `.table-bordered` class, you will get borders surrounding every element and rounded corners around the entire table

Hover Table

By adding the `.table-hover` class, a light gray background will be added to rows while the cursor hovers over them

Condensed Table

By adding the `.table-condensed` class, row padding is cut in half to condense the table.

Contextual classes

The Contextual classes shown in following table will allow you to change the background color of your table rows or individual cells.

Class	Description
.active	Applies the hover color to a particular row or cell
.success	Indicates a successful or positive action
.warning	Indicates a warning that might need attention
.danger	Indicates a dangerous or potentially negative action

These classes can be applied to `<tr>`, `<td>` or `<th>`.

Responsive Tables

By wrapping any `.table` in `.table-responsive` class, you will make the table scroll horizontally up to small devices (under 768px). When viewing on anything larger than 768px wide, you will not see any difference in these tables.

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

    <caption>Responsive Table Layout</caption>

    <thead>
      <tr>
        <th>Product</th>
        <th>Payment Date</th>
        <th>Status</th>
      </tr>
    </thead>

    <tbody>
      <tr>
        <td>Product1</td>
        <td>23/11/2013</td>
        <td>Pending</td>
      </tr>

      <tr>
        <td>Product2</td>
        <td>10/11/2013</td>
        <td>Delivered</td>
      </tr>

      <tr>
        <td>Product3</td>
        <td>20/10/2013</td>
        <td>In Call to confirm</td>
      </tr>
    </tbody>
  </table>
</div>
```

```
        <tr>
          <td>Product4</td>
          <td>20/10/2013</td>
          <td>Declined</td>
        </tr>
      </tbody>

    </table>
  </div>
```

Panels

Sometimes you might require to place your content in box for better presentation. In such situation the Bootstrap panel component can be very useful. In most basic form the panel component applies some border and padding around the content.

```
<div class="panel panel-default">
  <div class="panel-body">Look, I'm in a panel!</div>
</div>
```

Panels with Heading

You can also add a heading to your panel with .panel-heading class.

```
<div class="panel panel-default">
  <div class="panel-heading">This Page is Disabled</div>
  <div class="panel-body">This page is temporarily disabled by the site administrator for
some reason.<br> <a href="#">Click here</a> to enable the page.</div>
</div>
```

You can also include heading elements from <h1> to <h6> with a .panel-title class.

```
<div class="panel panel-default">
  <div class="panel-heading">
    <h1 class="panel-title">Panel Title</h1>
  </div>
  <div class="panel-body">Panel content...</div>
</div>
```

Panels with Footer

Like heading you can also add footer section to your panels using the .panel-footer class. The panel's footer can be used to wrap buttons or secondary text.

```
<div class="panel panel-default">
  <div class="panel-body">This page is temporarily disabled by the site administrator for
some reason.</div>
```



```

<div class="panel-footer clearfix">
  <div class="pull-right">
    <a href="#" class="btn btn-primary">Learn More</a>
    <a href="#" class="btn btn-default">Go Back</a>
  </div>
</div>
</div>

```

Panels with Contextual States

Like other component you also add contextual state classes like `.panel-primary`, `.panel-success`, `.panel-info`, `.panel-warning`, or `.panel-danger` on the panel components to make it more meaningful and drawing attention of the user

```

<div class="panel panel-primary">
  <div class="panel-heading">
    <h3 class="panel-title">301 Moved Permanently</h3>
  </div>
  <div class="panel-body">The requested page has been permanently moved to a new
location.</div>
</div>
<div class="panel panel-success">
  <div class="panel-heading">
    <h3 class="panel-title">200 OK</h3>
  </div>
  <div class="panel-body">The server successfully processed the request.</div>
</div>
<div class="panel panel-info">
  <div class="panel-heading">
    <h3 class="panel-title">100 Continue</h3>
  </div>
  <div class="panel-body">The client should continue with its request.</div>
</div>
<div class="panel panel-warning">
  <div class="panel-heading">
    <h3 class="panel-title">400 Bad Request</h3>
  </div>
  <div class="panel-body">The request cannot be fulfilled due to bad syntax.</div>
</div>
<div class="panel panel-danger">
  <div class="panel-heading">
    <h3 class="panel-title">503 Service Unavailable</h3>
  </div>
  <div class="panel-body">The server is temporarily unable to handle the request.</div>
</div>

```

Accordion

Creating Accordion Widget with Bootstrap

Accordion menus and widgets are widely used on the websites to manage the large amount of content and navigation lists. With Bootstrap collapse plugin you can either create accordion or a simple collapsible panel without writing any JavaScript code

```
<div id="accordion" class="panel-group">
  <div class="panel panel-default">
    <div class="panel-heading">
      <h4 class="panel-title">
        <a data-toggle="collapse" data-parent="#accordion" href="#collapseOne">1.
What is HTML?</a>
      </h4>
    </div>
    <div id="collapseOne" class="panel-collapse collapse">
      <div class="panel-body">
        <p>HTML stands for HyperText Markup Language. HTML is the main markup
language for describing the structure of Web pages. <a
href="https://www.tutorialrepublic.com/html-tutorial/" target="_blank">Learn more.</a></p>
      </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">2.
What is Bootstrap?</a>
      </h4>
    </div>
    <div id="collapseTwo" class="panel-collapse collapse in">
      <div class="panel-body">
        <p>Bootstrap is a powerful front-end framework for faster and easier web
development. It is a collection of CSS and HTML conventions. <a
href="https://www.tutorialrepublic.com/twitter-bootstrap-tutorial/" target="_blank">Learn
more.</a></p>
      </div>
    </div>
  </div>
  <div class="panel panel-default">
    <div class="panel-heading">
      <h4 class="panel-title">
        <a data-toggle="collapse" data-parent="#accordion" href="#collapseThree">3.
What is CSS?</a>
      </h4>
    </div>
    <div id="collapseThree" class="panel-collapse collapse">
      <div class="panel-body">
```

```

        <p>CSS stands for Cascading Style Sheet. CSS allows you to specify various
        style properties for a given HTML element such as colors, backgrounds, fonts etc. <a
        href="https://www.tutorialrepublic.com/css-tutorial/" target="_blank">Learn more.</a></p>
    </div>
</div>
</div>
</div>

```

Bootstrap Accordion with Plus Minus Icons

You can also add plus minus icons to the Bootstrap accordion widget to make it visually more attractive with a few lines of jQuery code, as follow:

```

<script type="text/javascript">
$(document).ready(function(){
    // Add minus icon for collapse element which is open by default
    $(".collapse.in").each(function(){
        $(this).siblings(".panel-heading").find(".glyphicon").addClass("glyphicon-
minus").removeClass("glyphicon-plus");
    });

    // Toggle plus minus icon on show hide of collapse element
    $(".collapse").on('show.bs.collapse', function(){
        $(this).parent().find(".glyphicon").removeClass("glyphicon-
plus").addClass("glyphicon-minus");
    }).on('hide.bs.collapse', function(){
        $(this).parent().find(".glyphicon").removeClass("glyphicon-
minus").addClass("glyphicon-plus");
    });
});
</script>

```

The `show.bs.collapse` and `hide.bs.collapse` in the example above are collapse events, you'll learn about the events little later in this chapter.

Expanding and Collapsing Elements via Data Attributes

You can use the Bootstrap collapse feature for expanding and collapsing any specific element via data attributes without using the accordion markup

```

<!-- Trigger Button HTML -->

<input type="button" class="btn" data-toggle="collapse" data-target="#toggleDemo"
value="Toggle Button">

<!-- Collapsible Element HTML -->

<div id="toggleDemo" class="collapse in"><p>This is a simple example of expanding and
collapsing individual element via data attribute. Click on the <b>Toggle Button</b> button
to see the effect.</p></div>

```

Explanation of Code

- The Bootstrap collapse plugin basically requires the two elements to work properly — the controller element such as a button or hyperlink by clicking on which you want to collapse the other element, and the collapsible element itself.
- The `data-toggle="collapse"` attribute (line no-2) is added to the controller element along with an attribute `data-target` (for buttons) or `href` (for anchors) to automatically assign the control of a collapsible element.
- The `data-target` or `href` attribute accepts a CSS selector to apply the collapse to a specific element. Be sure to add the class `.collapse` to the collapsible element.
- You can optionally add the class `.in` to the collapsible element in addition to the class `.collapse` to make it open by default.