BOOTSTRAP QUICK GUIDE

BOOTSTRAP OVERVIEW

What is Twitter Bootstrap?

Bootstrap is a sleek, intuitive, and powerful, mobile first front-end framework for faster and easier web development. It uses HTML, CSS and Javascript.

History

Bootstrap was developed by *Mark Otto* and *Jacob Thornton* at *Twitter*. It was released as an open source product in August 2011 on GitHub.

Why Use Bootstrap?

- **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. More about the responsive design is in the chapter Bootstrap Responsive Design.

- 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.

What Bootstrap Package Includes?

- **Scaffolding** : Bootstrap provides a basic structure with Grid System, link styles, and background. This is is covered in detail in the section **Bootstrap Basic Structure**
- **CSS** : Bootstrap comes with the feature of global CSS settings, fundamental HTML elements styled and enhanced with extensible classes, and an advanced grid system. This is covered in detail in the section **Bootstrap with CSS**.
- **Components** : Bootstrap contains over a dozen reusable components built to provide iconography, dropdowns, navigation, alerts, pop-overs, and much more. This is covered in detail in the section **Layout Components**.

- **JavaScript Plugins** : Bootstrap contains over a dozen custom jQuery plugins. You can easily include them all, or one by one. This is covered in details in the section **Bootstrap Plugins**.
- **Customize** : You can customize Bootstrap's components, LESS variables, and jQuery plugins to get your very own version.

BOOTSTRAP ENVIRONMENT SETUP

It is very easy to setup and start using Bootstrap. This chapter will explain how to download and setup Bootstrap. We will also discuss the Bootstrap file structure, and demonstrate its usage with an example.

Download Bootstrap

You can download the latest version of Bootstrap from http://getbootstrap.com/. When you click on this link, you will get to see a screen as below:

Here you can see two buttons:

- *Download Bootstrap* : Clicking this, you can download the precompiled and minified versions of Bootstrap CSS, JavaScript, and fonts. No documentation or original source code files are included.
- *Download Source* : Clicking this, you can get the latest Bootstrap LESS and JavaScript source code directly from GitHub.

> *If you work with Bootstrap's uncompiled source code, you need to compile the LESS files to produce usable CSS files. For compiling LESS files into CSS, Bootstrap officially supports only Recess , which is Twitter's CSS hinter based on less.js.*

For better understanding and ease of use, we shall use precompiled version of Bootstrap throughout the tutorial. As the files are complied and minified you don't have to bother every time including separate files for individual functionality. At the time of writing this tutorial the latest version *Bootstrap*3 was downloaded.

File structure

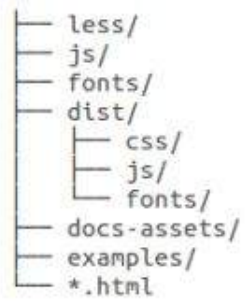
Precompiled Bootstrap

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 Source Code

If you have downloaded the Bootstrap source code then the file structure would be as follows:



- The files under *less/*, *js/*, and *fonts/* are the source code for Bootstrap CSS, JS, and icon fonts *respectively*.
- The *dist/* folder includes everything listed in the precompiled download section above.
- *docs-assets/*, *examples/*, and all **.html* files are Bootstrap documentation.

HTML Template

A basic HTML template using Bootstrap would look like this:

```
<!DOCTYPE html>
<html>
   <head>
      <title>Bootstrap 101 Template</title>
      <meta name="viewport" content="width=device-width, initial-scale=1.0">
      <!-- Bootstrap -->
      <link href="css/bootstrap.min.css" rel="stylesheet">

      <!-- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media
         queries -->
      <!-- WARNING: Respond.js doesn't work if you view the page
         via file:// -->
      <!--[if lt IE 9]>
         <script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/
            html5shiv.js"></script>
         <script src="https://oss.maxcdn.com/libs/respond.js/1.3.0/
            respond.min.js"></script>
      <![endif]-->
   </head>
   <body>
      <h1>Hello, world!</h1>

      <!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
      <script src="https://code.jquery.com/jquery.js"></script>
      <!-- Include all compiled plugins (below), or include individual files
            as needed -->
      <script src="js/bootstrap.min.js"></script>
   </body>
</html>
```

Here you can see the **jquery.js**, **bootstrap.min.js** and **bootstrap.min.css** files that are included to make a normal HTM file to the Bootstrapped Template. Just make sure to include jQuery library before you include Bootstrap library.

More details about each of the elements in this above piece of code will be discussed in the chapter Bootstrap CSS Overview.

> *This template structure is already included as part of the* ***Try it*** *onlinecompiler tool. Hence in all the examples inthefollowingchapters of this tutorial you will only see the*

> *contents of the <body> element. Once you click on the **Try it** option available at the top right corner of example, and you will see the entire code.*

Example

Now let's try an example using the above template. Try the following example using Try it option available at the top right corner of the below sample code box on our website:

```
<h1>Hello, world!</h1>
```

> *In all the subsequent chapters we have used dummy text from the site http://www.lipsum.com/.*

BOOTSTRAP GRID SYSTEM

In this chapter we shall discuss the Bootstrap Grid System.

What is a Grid?

As put by wikepedia:

> *In graphic design, a grid is a structure usuallytwo − 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.*

To put in simple words, grids in web design organise and structure content, makes the websites easy to scan and reduces the cognitive load on users.

What is Bootstrap Grid System?

As put by the official documentation of Bootstrap for grid system:

> *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.*

Let us understand the above statement. Bootstrap 3 is mobile first in the sense that the code for Bootstrap now starts by targeting smaller screens like mobile devices, tablets, and then “expands” components and grids for larger screens such as laptops, desktops.

Mobile First Strategy

- **Content**
  - Determine what is most important.
- **Layout**
  - Design to smaller widths first.
  - Base CSS address mobile device first; media queries address for tablet, desktops.
- **Progressive Enhancement**
  - Add elements as screen size increases.

Working of Bootstrap Grid System

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 *gapsbetweencolumncontent* 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**.

Media Queries

Media query is a really fancy term for "conditional CSS rule". It simply applies some CSS, based on certain conditions set forth. If those conditions are met, the style is applied.

Media Queries in Bootstrap allow you to move, show and hide content based on the viewport size. Following media queries are used in LESS files to create the key breakpoints in the Bootstrap grid system.

```
/* Extra small devices (phones, less than 768px) */
/* No media query since this is the default in Bootstrap */

/* Small devices (tablets, 768px and up) */
@media (min-width: @screen-sm-min) { ... }

/* Medium devices (desktops, 992px and up) */
@media (min-width: @screen-md-min) { ... }

/* Large devices (large desktops, 1200px and up) */
@media (min-width: @screen-lg-min) { ... }
```

Occasionally these are expanded to include a **max-width** to limit CSS to a narrower set of devices.

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

Media queries have two parts, a device specification and then a size rule. In the above case, the following rule is set:

Let us consider this line:

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

For all devices no matter what kind with *min-width: @screen-sm-min* if the width of the screen gets smaller than *@screen-sm-max*, *then do something*.

Grid options

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

	Extra small devices Phones $< 768px$	**Small devices Tablets** $\geq 768px$	**Medium devices Desktops** $\geq 992px$	**Large devices Desktops** $\geq 1200px$
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 15*px*on*e*a*ch*si*de*of*a*co*l*umn*	30px 15*px*on*e*a*ch*si*de*of*a*co*l*umn*	30px 15*px*on*e*a*ch*si*de*of*a*co*l*umn*	30px 15*px*on*e*a*ch*si*de*of*a*co*l*umn*
Nestable	Yes	Yes	Yes	Yes
Offsets	Yes	Yes	Yes	Yes
Column ordering	Yes	Yes	Yes	Yes

Basic Grid Structure

Following is basic structure of Bootstrap grid:

```
<div >
   <div >
      <div ></div>
      <div ></div>
   </div>
   <div >...</div>
</div>
<div >....
```

Let us see some simple grid examples:

- **Example: Stacked-to-horizontal**
- **Example: Medium and Large Device**
- **Example: Mobile, tablet, desktops**

Responsive column resets

With the four tiers of grids available, you are bound to run into issues where at certain breakpoints, the columns don't clear quite right as one is taller than the other. To fix that, use a combination of a class **.clearfix** and the responsive utility classes as shown in the following example:

```
<div >
   <div  >
      <div
         style="background-color: #dedef8;
         box-shadow: inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
         <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit.</p>
      </div>
      <div
         style="background-color: #dedef8;box-shadow:
         inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
         <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
            eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut
            enim ad minim veniam, quis nostrud exercitation ullamco laboris
            nisi ut aliquip ex ea commodo consequat.
         </p>
         <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
            eiusmod tempor incididunt ut.
         </p>
```

```
      </div>

      <div ></div>

      <div
         style="background-color: #dedef8;
         box-shadow:inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
         <p>Ut enim ad minim veniam, quis nostrud exercitation ullamco
            laboris nisi ut aliquip ex ea commodo consequat.
         </p>
      </div>
      <div
         style="background-color: #dedef8;box-shadow:
         inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
         <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
            eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut
            enim ad minim
         </p>
      </div>
   </div>
</div>
```

> *Resize your viewport or check it out on your phone for a desired result of this example.*

Offset Columns

Offsets are a useful feature for more specialized layouts. They can be used to push columns over for more spacing, *forexample*. The **.col-xs=*** classes don't support offsets, but they are easily replicated by using an empty cell.

To use offsets on large displays, use the **.col-md-offset-*** classes. These classes increase the left margin of a column by * columns where * range from **1** to **11**.

In the following example, we have .. , We will center this using class **.col-md-offset-3**.

```
<div >

   <h1>Hello, world!</h1>

   <div  >
      <div
         style="background-color: #dedef8;box-shadow:
         inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
         <p>Lorem ipsum dolor sit amet, consectetur adipisicing
            elit.
         </p>
      </div>

   </div>
</div>
```

Nesting columns

To nest your content with the default grid, add a new **.row** and set of **.col-md-*** columns within an existing **.col-md-*** column. Nested rows should include a set of columns that add up to 12.

In the following example, the layout has two columns, with the second one being split into four boxes over two rows.

```
<div >

   <h1>Hello, world!</h1>
```

```
   <div >

      <div background-color: #dedef8;box-shadow:
         inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
         <h4>First Column</h4>
         <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit.</p>
      </div>

      <div background-color: #dedef8;box-shadow:
         inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
         <h4>Second Column- Split into 4 boxes</h4>
         <div >
            <div background-color: #B18904;
               box-shadow: inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
               <p>Consectetur art party Tonx culpa semiotics. Pinterest
                  assumenda minim organic quis.
               </p>
            </div>
            <div background-color: #B18904;
               box-shadow: inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
               <p> sed do eiusmod tempor incididunt ut labore et dolore magna
                  aliqua. Ut enim ad minim veniam, quis nostrud exercitation
                  ullamco laboris nisi ut aliquip ex ea commodo consequat.
               </p>
            </div>
         </div>

         <div >
            <div background-color: #B18904;
               box-shadow: inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
               <p>quis nostrud exercitation ullamco laboris nisi ut
                  aliquip ex ea commodo consequat.
               </p>
            </div>
            <div background-color: #B18904;
               box-shadow: inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
               <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit,
                  sed do eiusmod tempor incididunt ut labore et dolore magna
                  aliqua. Ut enim ad minim.</p>
            </div>
         </div>

      </div>

   </div>

</div>
```

Column Ordering

Another nice feature of Bootstrap grid system is that you can easily write the columns in an order, and show them in another one. You can easily change the order of built-in grid columns with **.col-md-push-*** and **.col-md-pull-*** modifier classes where * range from **1** to **11**.

In the following example we have two columns layout with left column being the narrowest and acting as a sidebar. We will swap the order of these columns using **.col-md-push-*** and **.col-md-pull-*** classes.

```
<div >

   <h1>Hello, world!</h1>

   <div >
      <p>Before Ordering</p>
      <div background-color: #dedef8;
```

```
         box-shadow: inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
         I am on left
      </div>
      <div background-color: #dedef8;
         box-shadow: inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
         I am on right
      </div>
   </div><br>
   <div >
      <p>After Ordering</p>
      <div
         style="background-color: #dedef8;
         box-shadow: inset 1px -1px 1px #444,
         inset -1px 1px 1px #444;">
         I was on left
      </div>
      <div
         style="background-color: #dedef8;
         box-shadow: inset 1px -1px 1px #444,
         inset -1px 1px 1px #444;">
         I was on right
      </div>
   </div>

</div>
```

BOOTSTRAP CSS OVERVIEW

This chapter provides an overview of the key pieces of Bootstrap's infrastructure, including Bootstrap's approach to better, faster, stronger web development.

HTML5 doctype

Bootstrap makes use of certain HTML elements and CSS properties that require the use of the HTML5 doctype. Hence include the below piece of code for HTML5 doctype at the beginning of all your projects using Bootstrap.

```
<!DOCTYPE html>
<html>
....
</html>
```

Mobile First

Since Bootstrap 3 has been launched, Bootstrap has become mobile first. It means 'mobile first' styles can be found throughout the entire library instead of them in separate files. 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.

Add **user-scalable=no** to the **content** attribute to disable zooming capabilities on mobile devices as shown below. Users are only able to scroll and not zoom with this change, and results in your site feeling a bit more like a native application.

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

```
                                        user-scalable=no">
```

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

Responsive Images

Bootstrap 3 allows you to make the images responsive by adding a class **.img-responsive** to the **<img>** tag. This class applies **max-width: 100%;** and **height: auto;** to the image so that it scales nicely to the parent element.

```
<img src="..." >
```

Typography and Links

Bootstrap sets a basic global display *background*, typography, and link styles:

- **Basic Global display** : Sets *background-color: #fff;* on the *<body>* element.
- **Typography** : Uses the *@font-family-base*, *@font-size-base*, and *@line-height-base* attributes as the typographic base.
- **Link styles** : Sets the global link color via attribute *@link-color* and apply link underlines only on *:hover*.

> *If you intend to use LESS code, you may find all these within scaffolding.less.*

Normalize

Bootstrap uses Normalize to establish cross browser consistency.

Normalize.css is a modern, HTML5-ready alternative to CSS resets. It is a small CSS file that provides better cross-browser consistency in the default styling of HTML elements.

Containers

Use class **.container** to wrap a page's content and easily center the content's as shown below.

```
<div >
  ...
</div>
```

Take a look at the **.container** class in *bootstrap.css* file:

```
.container {
   padding-right: 15px;
   padding-left: 15px;
   margin-right: auto;
   margin-left: auto;
}
```

Note that, due to padding and fixed widths, containers are not nestable by default.

Take a look at *bootstrap.css* file:

```
@media (min-width: 768px) {
   .container {
      width: 750px;
}
```

Here you can see that CSS has media-queries for containers with **width**. This helps for applying responsiveness and within those the container class is modified accordingly to render the grid system properly.

BOOTSTRAP TYPOGRAPHY

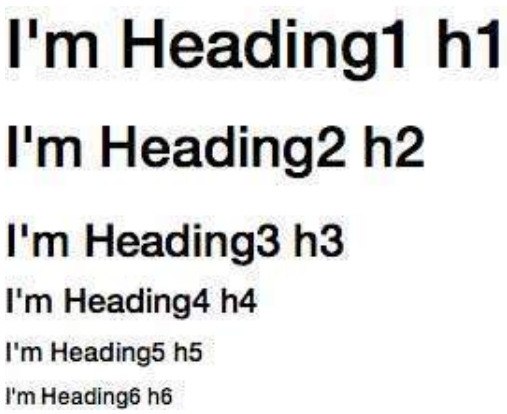
Bootstrap uses Helvetica Neue, Helvetica, Arial, and sans-serif in its default font stack. Using typography feature of Bootstrap you can create headings, paragraphs, lists and other inline elements. Let see learn each one of these in the following sections.

Headings

All HTML headings *h1toh6* 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>
```

The above code segment with Bootstrap will produce following result:

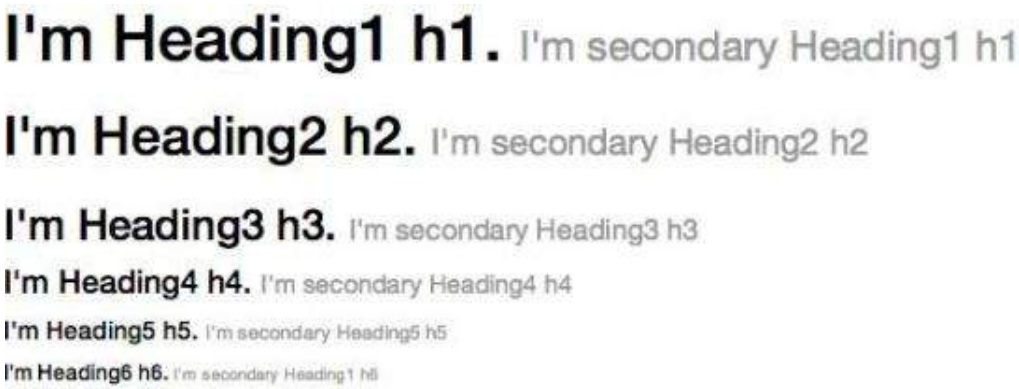


Inline Subheadings

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>
```

The above code segment with Bootstrap will produce following result:

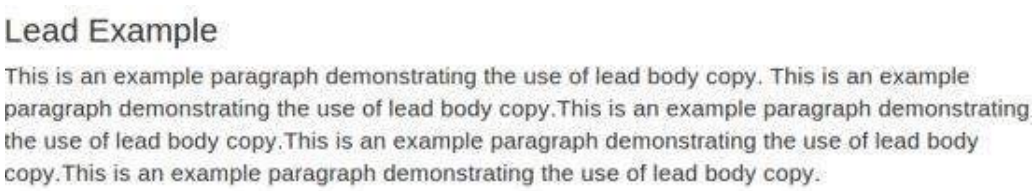


Lead Body Copy

To add some emphasis to a paragraph, add . 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 >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>
```

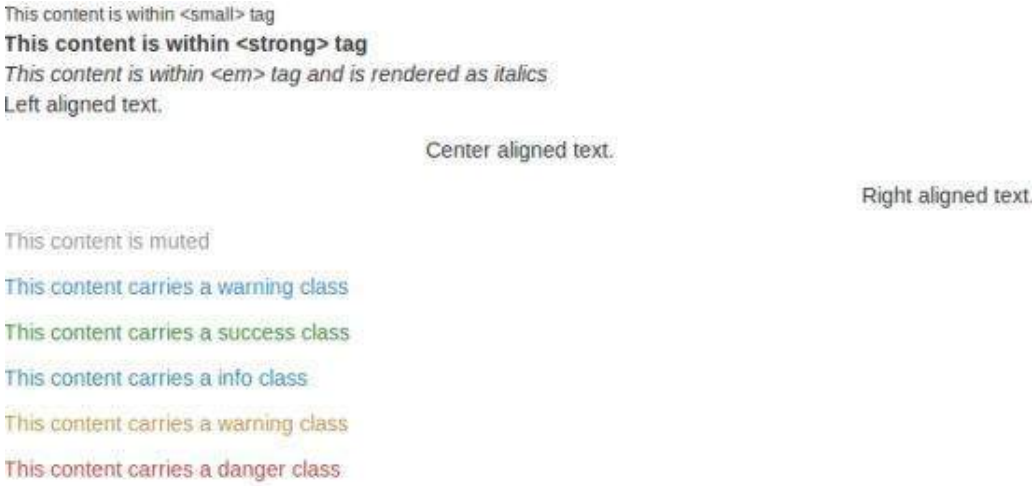


Emphasis

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 <small> tag</small><br>
<strong>This content is within <strong> tag</strong><br>
<em>This content is within <em> tag and is rendered as italics</em><br>
<p >Left aligned text.</p>
<p >Center aligned text.</p>
<p >Right aligned text.</p>
<p >This content is muted</p>
<p >This content carries a primary class</p>
<p >This content carries a success class</p>
<p >This content carries a info class</p>
<p >This content carries a warning class</p>
<p >This content carries a danger class</p>
```



Abbreviations

The HTML 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 *aslongasyouaddthattexttothe < abbr > titleattribute*. To get a a slightly smaller font size add .initialism to <abbr>.

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

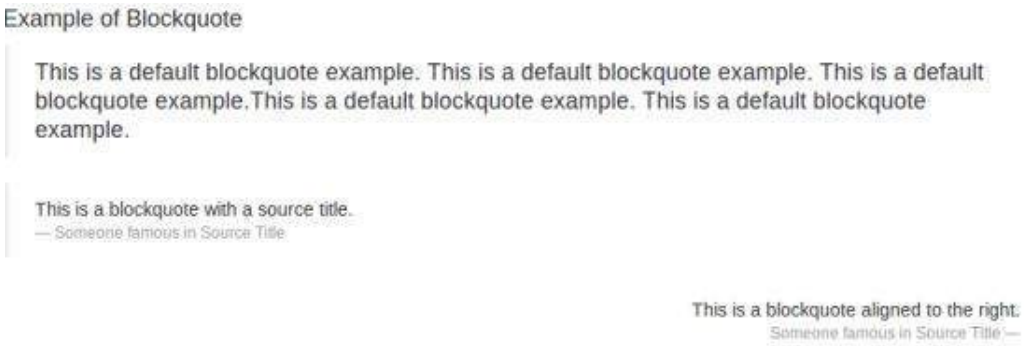
```
<address>
  <strong>Some Company, Inc.</strong><br>
  007 street<br>
  Some City, State XXXXX<br>
  <abbr title="Phone">P:</abbr> (123) 456-7890
</address>

<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 >Source Title</cite></small></blockquote>
```



Lists

Bootstrap supports ordered lists, unordered lists, and definition lists.

- **Ordered lists** : An ordered list is a list that falls in some sort of sequential order and is prefaced by numbers.
- **Unordered lists** : An unordered list is a list that doesn't have any particular order and is traditionally styled with bullets. If you do not want the bullets to appear, then you can remove the styling by using the class *.list-unstyled*. You can also place all list items on a single line using the class *.list-inline*.
- **Definition lists**: In this type of list, each list item can consist of both the <dt> and the <dd> elements. <dt> stands for *definition term*, and like a dictionary, this is the term *orphrase* that is being defined. Subsequently, the <dd> is the definition of the <dt>.

  You can make terms and descriptions in <dl> line up side-by-side using class *dl-horizontal*.

The following example demonstrates each of these types:

```
<h4>Example of Ordered List</h4>
<ol>
```

```
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
  <li>Item 4</li>
</ol>
<h4>Example of UnOrdered List</h4>
<ul>
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
  <li>Item 4</li>
</ul>
<h4>Example of Unstyled List</h4>
<ul >
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
  <li>Item 4</li>
</ul>
<h4>Example of Inline List</h4>
<ul >
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
  <li>Item 4</li>
</ul>
<h4>Example of Definition List</h4>
<dl>
  <dt>Description 1</dt>
  <dd>Item 1</dd>
  <dt>Description 2</dt>
  <dd>Item 2</dd>
</dl>
<h4>Example of Horizontal Definition List</h4>
<dl >
  <dt>Description 1</dt>
  <dd>Item 1</dd>
  <dt>Description 2</dt>
  <dd>Item 2</dd>
</dl>
```

Example of Ordered List

1. Item 1
2. Item 2
3. Item 3
4. Item 4

Example of UnOrdered List

- Item 1
- Item 2
- Item 3
- Item 4

Example of Unstyled List

Item 1
Item 2
Item 3
Item 4

Example of Inline List

Item 1 Item 2 Item 3 Item 4

Example of Definition List

Description 1
Item 1
Description 2

Item 2

Example of Horizontal Definition List

Description 1 Item 1
Description 2 Item 2

BOOTSTRAP CODE

Bootstrap allows you to display code with two different key ways:

- The first is the <code> tag. If you are going to be displaying code inline, you should use the <code> tag.
- Second is the <pre> tag. If the code needs to be displayed as a standalone block element or if it has multiple lines, then you should use the <pre> tag.

> *Make sure that when you use the <pre> and <code> tags, you use the unicode variants for the opening and closing tags:* ***<*** *and* ***>****.*

Let us see an example below:

```
<p><code>&lt;header&gt;</code> is wrapped as an inline element.</p>
<p>To display code as a standalone block element use &lt;pre&gt; tag as:
<pre>
   &lt;article&gt;
      &lt;h1&gt;Article Heading&lt;/h1&gt;
   &lt;/article&gt;
</pre>
```

<header> is wrapped as an inline element.

To display code as a standalone block element use <pre> tag as:

```
<article>
<h1>Article Heading</h1>
</article>
```

BOOTSTRAP TABLES

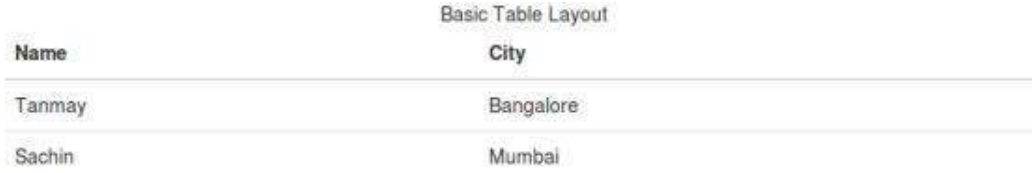
Bootstrap provides a clean layout for building tables. Some of the table elements supported by Bootstrap are:

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 *orrow*, *dependingonscopeandplacement* 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 >
   <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> as seen in the following example:

```
<table >
   <caption>Striped Table Layout</caption>
   <thead>
      <tr>
         <th>Name</th>
         <th>City</th>
         <th>Pincode</th>
      </tr>
   </thead>
   <tbody>
      <tr>
         <td>Tanmay</td>
         <td>Bangalore</td>
         <td>560001</td>
      </tr>
      <tr>
         <td>Sachin</td>
         <td>Mumbai</td>
         <td>400003</td>
      </tr>
      <tr>
         <td>Uma</td>
         <td>Pune</td>
         <td>411027</td>
      </tr>
   </tbody>
</table>
```

Tanmay	Bangalore	560001
Sachin	Mumbai	400003
Uma	Pune	411027

Bordered Table

By adding the *.table-bordered* class, you will get borders surrounding every element and rounded corners around the entire table as seen in the following example:

```
<table >
   <caption>Bordered Table Layout</caption>
   <thead>
      <tr>
         <th>Name</th>
         <th>City</th>
         <th>Pincode</th>
      </tr>
   </thead>
   <tbody>
      <tr>
         <td>Tanmay</td>
         <td>Bangalore</td>
         <td>560001</td>
      </tr>
      <tr>
         <td>Sachin</td>
         <td>Mumbai</td>
         <td>400003</td>
      </tr>
      <tr>
         <td>Uma</td>
         <td>Pune</td>
         <td>411027</td>
      </tr>
   </tbody>
</table>
```

Bordered Table Layout

Name	City	Pincode
Tanmay	Bangalore	560001
Sachin	Mumbai	400003
Uma	Pune	411027

Hover Table

By adding the *.table-hover* class, a light gray background will be added to rows while the cursor hovers over them, as seen in the following example:

```
<table >
   <caption>Hover Table Layout</caption>
   <thead>
      <tr>
         <th>Name</th>
         <th>City</th>
         <th>Pincode</th>
      </tr>
   </thead>
   <tbody>
      <tr>
         <td>Tanmay</td>
         <td>Bangalore</td>
         <td>560001</td>
      </tr>
      <tr>
         <td>Sachin</td>
```

```
         <td>Mumbai</td>
         <td>400003</td>
      </tr>
      <tr>
         <td>Uma</td>
         <td>Pune</td>
         <td>411027</td>
      </tr>
   </tbody>
</table>
```

Hover Table Layout

Name	City	Pincode
Tanmay	Bangalore	560001
Sachin	Mumbai	400003
Uma	Pune	411027

Condensed Table

By adding the *.table-condensed* class, row padding is cut in half to condense the table. as seen in the following example. This is useful if you want any denser information.

```
<table >
   <caption>Condensed Table Layout</caption>
   <thead>
      <tr>
         <th>Name</th>
         <th>City</th>
         <th>Pincode</th>
      </tr>
   </thead>
   <tbody>
      <tr>
         <td>Tanmay</td>
         <td>Bangalore</td>
         <td>560001</td>
      </tr>
      <tr>
         <td>Sachin</td>
         <td>Mumbai</td>
         <td>400003</td>
      </tr>
      <tr>
         <td>Uma</td>
         <td>Pune</td>
         <td>411027</td>
      </tr>
   </tbody>
</table>
```

Condensed Table Layout

Name	City	Pincode
Tanmay	Bangalore	560001
Sachin	Mumbai	400003
Uma	Pune	411027

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>.

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

Contextual Table Layout

Product	Payment Date	Status
Product1	23/11/2013	Pending
Product2	10/11/2013	Delivered
Product3	20/10/2013	In Call to confirm
Product4	20/10/2013	Declined

Responsive tables

By wrapping any *.table* in *.table-responsive* class, you will make the table scroll horizontally up to small devices *under768px*. When viewing on anything larger than 768px wide, you will not see any difference in these tables.

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

Responsive Table Layout

Product	Payment Date	Status
Product1	23/11/2013	Pending
Product2	10/11/2013	Delivered
Product3	20/10/2013	In Call to confirm
Product4	20/10/2013	Declined

BOOTSTRAP FORMS

In this chapter, we will study how to create forms with ease using Bootstrap. 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:

- Vertical *default* form
- Inline form
- 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 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 >
      <label for="name">Name</label>
      <input type="text"
         placeholder="Enter Name">
   </div>
   <div >
      <label for="inputfile">File input</label>
      <input type="file" >
```

```
      <p >Example block-level help text here.</p>
   </div>
   <div >
      <label>
      <input type="checkbox"> Check me out
      </label>
   </div>
   <button type="submit" >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 >
   <div >
      <label >Name</label>
      <input type="text"
         placeholder="Enter Name">
   </div>
   <div >
      <label >File input</label>
      <input type="file" >
   </div>
   <div >
      <label>
      <input type="checkbox"> Check me out
      </label>
   </div>
   <button type="submit" >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 with class *.form-group*.
- Add a class of *.control-label* to the labels.

```
<form >
   <div >
      <label for="firstname" >First Name</label>
```

```
      <div >
         <input type="text"
            placeholder="Enter First Name">
      </div>
   </div>
   <div >
      <label for="lastname" >Last Name</label>
      <div >
         <input type="text"
            placeholder="Enter Last Name">
      </div>
   </div>
   <div >
      <div >
         <div >
            <label>
               <input type="checkbox"> Remember me
            </label>
         </div>
      </div>
   </div>
   <div >
      <div >
         <button type="submit" >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 >
    <label for="name">Label</label>
    <input type="text" >
  </div>
 </form>
```

Textarea

The textarea is used when you need multiple lines of input. Change *rows* attribute as necessary *fewerrows = smallerbox, morerows = biggerbox*.

```
<form role="form">
  <div >
    <label for="name">Text Area</label>
    <textarea ></textarea>
  </div>
</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 *defaultandinline* types:

```
<label for="name">Example of Default Checkbox and radio button </label>
<div >
   <label><input type="checkbox" value="">Option 1</label>
</div>
<div >
   <label><input type="checkbox" value="">Option 2</label>
</div>

<div >
   <label>
      <input type="radio" name="optionsRadios"
         value="option1" checked> Option 1
   </label>
</div>
<div >
   <label>
      <input type="radio" name="optionsRadios"
         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 >
      <input type="checkbox" > Option 1
   </label>
   <label >
      <input type="checkbox" > Option 2
   </label>
   <label >
      <input type="checkbox" > Option 3
   </label>
   <label >
      <input type="radio" name="optionsRadiosinline"
         value="option1" checked> Option 1
   </label>
   <label >
      <input type="radio" name="optionsRadiosinline"
         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 *selectandmultiple* types:

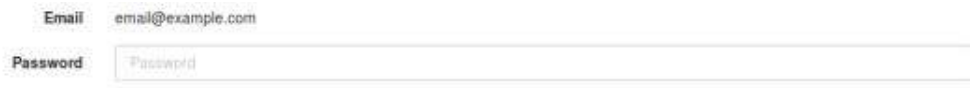
```
<form role="form">
   <div >
      <label for="name">Select list</label>
      <select >
         <option>1</option>
         <option>2</option>
         <option>3</option>
         <option>4</option>
         <option>5</option>
      </select>

      <label for="name">Mutiple Select list</label>
      <select multiple >
         <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 >
  <div >
    <label >Email</label>
    <div >
      <p >email@example.com</p>
    </div>
  </div>
  <div >
    <label for="inputPassword" >Password</label>
    <div >
      <input type="password"
         placeholder="Password">
    </div>
  </div>
</form>
```



Form Control States

In addition to the *:focus* *i. e.* , *auserclicksintotheinputortabsontoit* 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 >
   <div >
      <label >Focused</label>
      <div >
         <input
            value="This is focused...">
      </div>
   </div>
   <div >
      <label for="inputPassword" >
         Disabled
      </label>
      <div >
         <input
            placeholder="Disabled input here..." disabled>
      </div>
   </div>
   <fieldset disabled>
      <div >
         <label for="disabledTextInput"  >
            Disabled input (Fieldset disabled)
         </label>
         <div >
            <input type="text"
               placeholder="Disabled input">
         </div>
      </div>
      <div >
         <label for="disabledSelect"  >
            Disabled select menu (Fieldset disabled)
         </label>
         <div >
            <select  >
               <option>Disabled select</option>
            </select>
         </div>
      </div>
   </fieldset>
   <div >
      <label >
         Input with success
      </label>
      <div >
         <input type="text" >
      </div>
   </div>
   <div >
      <label >
         Input with warning
      </label>
      <div >
         <input type="text" >
      </div>
   </div>
```

```
   <div >
      <label >
         Input with error
      </label>
      <div >
         <input type="text" >
      </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 >
      <input
         placeholder=".input-lg">
   </div>

   <div >
      <input >
   </div>

   <div >
      <input
         placeholder=".input-sm">
   </div>
   <div >
   </div>
   <div >
      <select >
         <option value="">.input-lg</option>
      </select>
   </div>
   <div >
      <select >
         <option value="">Default select</option>
      </select>
   </div>
   <div >
      <select >
         <option value="">.input-sm</option>
      </select>
   </div>

   <div >
      <div >
         <input type="text" >
      </div>
      <div >
         <input type="text" >
      </div>
      <div >
         <input type="text" >
      </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 >
   <span >A longer block of help text that
   breaks onto a new line and may extend beyond one line.</span>
</form>
```

BOOTSTRAP BUTTONS

This chapter covers the use age of Bootstrap button with examples. Anything that is given a class of **.btn** will inherit the default look of a gray button with rounded corners. However, Bootstrap provides some options to style buttons, which are summarized in the following table:

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.

The following example demonstrates all the above button classes:

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

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

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

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

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

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

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

Button Size

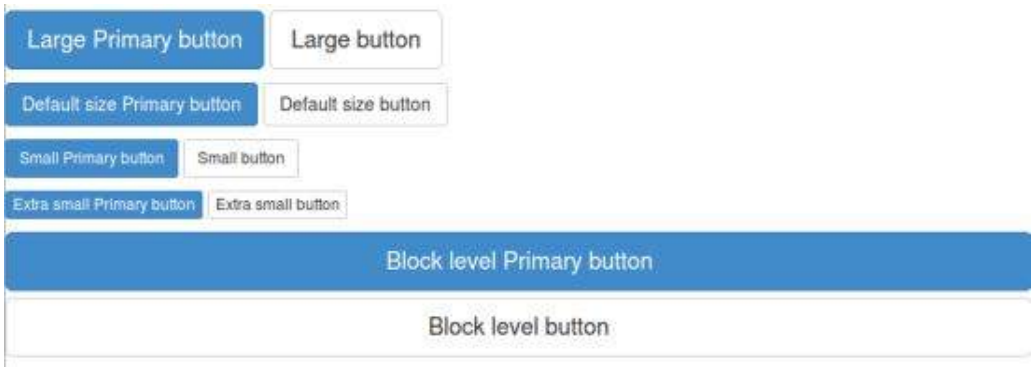
The following table summarizes the classes used to get buttons of various sizes:

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" >
      Large Primary button
   </button>
   <button type="button"
      >Large button
   </button>
</p>
<p>
   <button type="button" >
      Default size Primary button
   </button>
   <button type="button" >
      Default size button
   </button>
</p>
<p>
   <button type="button" >
      Small Primary button
   </button>
   <button type="button" >
      Small button
   </button>
</p>
<p>
   <button type="button" >
      Extra small Primary button
   </button>
   <button type="button" >
      Extra small button
   </button>
</p>
<p>
   <button type="button" >
```

```
      Block level Primary button
   </button>
   <button type="button" >
      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 *withadarkerbackground*, *darkerborder*, *andinsetshadow* 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" >
      Default Button
   </button>
   <button type="button" >
      Active Button
   </button>
</p>
<p>
   <button type="button" >
      Primary button
   </button>
   <button type="button" >
      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** *attribute* to <button> buttons.
Anchor element	Add the **disabled** *class* 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.*

The following example demonstrates this:

```
<p>
   <button type="button" >
      Default Button
   </button>
   <button type="button" >
      Disabled Button
   </button>
</p>
<p>
   <button type="button" >
      Primary button
   </button>
   <button type="button" >
      Disabled Primary button
   </button>
</p>
<p>
   <a href="#" >
      Link
   </a>
   <a href="#" >
      Disabled Link
   </a>
</p>
<p>
   <a href="#" >
      Primary link
   </a>
   <a href="#" >
      Disabled Primary link
   </a>
</p>
```

Button Tags

You may use button classes with <a>, <button>, or <input> element. But it is recommended that you use it with <button> elements mostly to avoid cross browser inconsistency issues.

The following example demonstrates this:

```
<a >Link</a>
<button >Button</button>
<input >
<input >
```

BOOTSTRAP IMAGES

This chapter covers the Bootstrap support for images. Bootstrap provides three classes that can be used to apply some simple styles to images:

- *.img-rounded* : adds *border-radius:6px* to give the image rounded corners.
- *.img-circle* : makes the entire image round by adding *border-radius:500px*.
- *.img-thumbnail* : adds a bit of padding and a gray border:

The following example demonstrates this:

```
<img src="/bootstrap/images/download.png"
   >
<img src="/bootstrap/images/download.png"
   >
<img src="/bootstrap/images/download.png"
   >
```

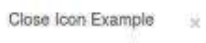

BOOTSTRAP HELPER CLASSES

This chapter discusses some of the helper classes in Bootstrap that might come in handy.

Close icon

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

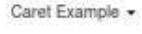
```
<p>Close Icon Example
   <button type="button" >
      &times;
   </button>
</p>
```



Carets

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

```
<p>Caret Example
   <span ></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 >
   Quick Float to left
</div>
<div >
   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 >
   <div >
      This is an example for center-block
   </div>
</div>
```

Clearfix

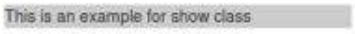
To clear the float of any element, use the **.clearfix** class.

```
<div >
   <div >
      Quick Float to left
   </div>
   <div >
      Quick Float to right
   </div>
</div>
```

Showing and Hiding Content

You can force an element to be shown or hidden *includingforscreenreaders* with the use of classes **.show** and **.hidden**.

```
<div >
   <div >
      This is an example for show class
   </div>
   <div >
      This is an example for hide class
   </div>
</div>
```



Screen Rader Content

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

```
<div >
```

```
   <form >
   <div >
      <label >Email address</label>
      <input type="email" >
   </div>
   <div >
      <label >Password</label>
      <input type="password" >
   </div>
</div>
```


Here we can see that the label of both the input types is assigned the class **sr-only**, hence labels will be visible to only screen readers.

BOOTSTRAP RESPONSIVE UTILITIES

Bootstrap provides some handful helper classes, for faster mobile-friendly development. These can be used for showing and hiding content by device via media query, combined with large, small, and medium devices.

Use these sparingly and avoid creating entirely different versions of the same site. **Responsive utilities are currently only available for block and table toggling**.

Classes	Devices
.visible-xs	Extra small *lessthan*768*px visible*
.visible-sm	Small *upto*768*px visible*
.visible-md	Medium 768*pxto*991*px visible*
.visible-lg	Larger 992*pxandabove visible*
.hidden-xs	Extra small *lessthan*768*px hidden*
.hidden-sm	Small *upto*768*px hidden*
.hidden-md	Medium 768*pxto*991*px hidden*
.hidden-lg	Larger 992*pxandabove hidden*

Print Classes

The following table lists the print classes. Use these for toggling the content for print.

Classes	Print
.visible-print	Yes Visible
.hidden-print	Visible only to browser not to print.

Example

The following example demonstrates the use of above listed helper classes. Resize your browser or load the example on different devices to test the responsive utility classes.

```
<div >
   <div >
      <div background-color: #dedef8;
         box-shadow: inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
         <span >Extra small</span>
         <span >✔ Visible on x-small</span>
      </div>
```

```
      <div background-color: #dedef8;
         box-shadow: inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
         <span >Small</span>
         <span >✔ Visible on small</span>
      </div>
      <div ></div>
      <div background-color: #dedef8;
         box-shadow: inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
         <span >Medium</span>
         <span >✔ Visible on medium</span>
      </div>
      <div background-color: #dedef8;
         box-shadow: inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
         <span >Large</span>
         <span >✔ Visible on large</span>
      </div>
</div>
```

Checkmarks indicates that the element is visible in your current viewport.

BOOTSTRAP GLYPHICONS

This chapter will discuss about Glyphicons, its use and some examples. Bootstrap bundles 200 glyphs in font format. Let us now understand what Glyphicons are.

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 glyphicons at this link http://tutorialspoint.com/bootstrap/bootstrap_glyph_icons.htm.

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 ></span>
```

The following example demonstrates this:

```
<p>
   <button type="button" >
      <span ></span>
   </button>
   <button type="button" >
```

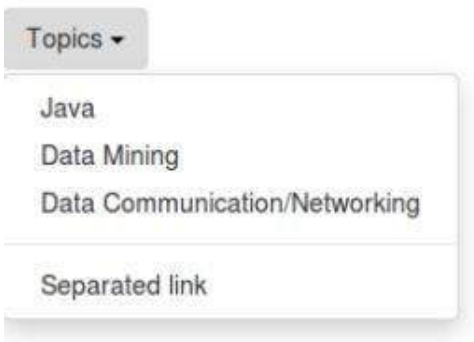
```
      <span ></span>
   </button>
   <button type="button" >
      <span ></span>
   </button>
   <button type="button" >
      <span ></span>
   </button>
</p>
<button type="button" >
  <span ></span> User
</button>
<button type="button" >
  <span ></span> User
</button>
<button type="button" >
  <span ></span> User
</button>
```

BOOTSTRAP DROPDOWNS

This chapter will highlight about Bootstrap dropdown menus. Dropdown menus are toggleable, contextual menus for displaying links in a list format. This can be made interactive with the dropdown JavaScript plugin.

To use dropdown, just wrap the dropdown menu within the class **.dropdown**.The following example demonstrates a basic dropdown menu:

```
<div >
   <button type="button"
      data-toggle="dropdown">
      Topics
      <span ></span>
   </button>
   <ul >
      <li role="presentation">
         <a role="menuitem" tabindex="-1" href="#">Java</a>
      </li>
      <li role="presentation">
         <a role="menuitem" tabindex="-1" href="#">Data Mining</a>
      </li>
      <li role="presentation">
         <a role="menuitem" tabindex="-1" href="#">
            Data Communication/Networking
         </a>
      </li>
      <li role="presentation" ></li>
      <li role="presentation">
         <a role="menuitem" tabindex="-1" href="#">Separated link</a>
      </li>
   </ul>
</div>
```



Options

Alignment

Align the dropdown menu to right by adding the class **.pull-right** to **.dropdown-menu**. The following example demonstrates this:

```
<div >
   <button type="button"
      data-toggle="dropdown">Topics
      <span ></span>
   </button>
   <ul
      aria-labelledby="dropdownMenu1">
      <li role="presentation">
         <a role="menuitem" tabindex="-1" href="#">Java</a>
      </li>
      <li role="presentation">
         <a role="menuitem" tabindex="-1" href="#">Data Mining</a>
      </li>
      <li role="presentation">
         <a role="menuitem" tabindex="-1" href="#">
            Data Communication/Networking
         </a>
      </li>
      <li role="presentation" ></li>
      <li role="presentation">
         <a role="menuitem" tabindex="-1" href="#">Separated link</a>
      </li>
   </ul>
</div>
```

Headers

You can add a header to label sections of actions in any dropdown menu by using the class **dropdown-header**. The following example demonstrates this:

```
<div >
   <button type="button"
      data-toggle="dropdown">
      Topics
      <span ></span>
   </button>
   <ul >
      <li role="presentation" >Dropdown header</li>
      <li role="presentation" >
         <a role="menuitem" tabindex="-1" href="#">Java</a>
      </li>
      <li role="presentation">
         <a role="menuitem" tabindex="-1" href="#">Data Mining</a>
      </li>
      <li role="presentation">
         <a role="menuitem" tabindex="-1" href="#">
            Data Communication/Networking
         </a>
      </li>
      <li role="presentation" ></li>
      <li role="presentation" >Dropdown header</li>
      <li role="presentation">
         <a role="menuitem" tabindex="-1" href="#">Separated link</a>
      </li>
   </ul>
```

```
</div>
```

BOOTSTRAP BUTTON GROUPS

Button groups allow multiple buttons to be stacked together on a single line. This is useful when you want to place items like alignment buttons together. You can add on optional JavaScript radio and checkbox style behavior with Bootstrap Button Plugin.

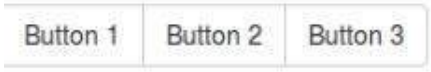
Following table summarizes the important classes Bootstrap provides to use button groups:

Class	Description	Code Sample
.btn-group	This class is used for a basic button group. Wrap a series of buttons with class **.btn** in **.btn-group**.	`<div class="btn-group"> <button type="button" >Button1</button> <button type="button" >Button2</button> </div>`
.btn-toolbar	This helps to combine sets of for more complex components.	`<div class="btn-toolbar" role="toolbar"> <div >...</div> <div >...</div> </div>`
.btn-group-lg, .btn-group-sm, .btn-group-xs	These classes can be applied to button group instead of resizing each button.	`<div class="btn-group btn-group-lg">...</div> <div >...</div> <div >...</div>`
.btn-group-vertical	This class make a set of buttons appear vertically stacked rather than horizontally.	`<div class="btn-group-vertical"> ... </div>`

Basic Button Group

The following example demonstrates the use of class **.btn-group** discussed in the above table:

```
<div >
  <button type="button" >Button 1</button>
  <button type="button" >Button 2</button>
  <button type="button" >Button 3</button>
</div>
```



Button Toolbar

The following example demonstrates the use of class **.btn-toolbar** discussed in the above table:

```
<div >
  <div >
  <button type="button" >Button 1</button>
  <button type="button" >Button 2</button>
  <button type="button" >Button 3</button>
 </div>
  <div >
  <button type="button" >Button 4</button>
  <button type="button" >Button 5</button>
  <button type="button" >Button 6</button>
  </div>
  <div >
  <button type="button" >Button 7</button>
  <button type="button" >Button 8</button>
  <button type="button" >Button 9</button>
  </div>
</div>
```

Button Size

The following example demonstrates the use of class **.btn-group-*** discussed in the above table:

```
<div >
  <button type="button" >Button 1</button>
  <button type="button" >Button 2</button>
  <button type="button" >Button 3</button>
 </div>
  <div >
  <button type="button" >Button 4</button>
  <button type="button" >Button 5</button>
  <button type="button" >Button 6</button>
</div>
  <div >
  <button type="button" >Button 7</button>
  <button type="button" >Button 8</button>
  <button type="button" >Button 9</button>
</div>
```

Nesting

You can nest button groups within another button group i.e, place a **.btn-group** within another **.btn-group** . This is done when you want dropdown menus mixed with a series of buttons.

```
<div >
  <button type="button" >Button 1</button>
  <button type="button" >Button 2</button>

  <div >
    <button type="button"
      data-toggle="dropdown">
      Dropdown
      <span ></span>
```

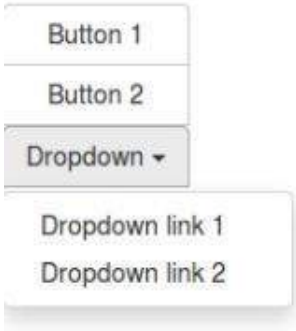
```
    </button>
    <ul >
      <li><a href="#">Dropdown link 1</a></li>
      <li><a href="#">Dropdown link 2</a></li>
    </ul>
  </div>
</div>
```

Vertical Buttongroup

The following example demonstrates the use of class **.btn-group-vertical** discussed in the above table:

```
<div >
  <button type="button" >Button 1</button>
  <button type="button" >Button 2</button>

  <div >
    <button type="button"
      data-toggle="dropdown">
      Dropdown
      <span ></span>
    </button>
    <ul >
      <li><a href="#">Dropdown link 1</a></li>
      <li><a href="#">Dropdown link 2</a></li>
    </ul>
  </div>
</div>
```



BOOTSTRAP BUTTON DROPDOWNS

This chapter will discuss about how to add dropdown menu to buttons using Bootstrap classes. To add a dropdown to a button, simply wrap the button and dropdown menu in a **.btn-group**. You can also use to act as an indicator that the button is a dropdown.

The following example demonstrates a basic single button dropdowns:

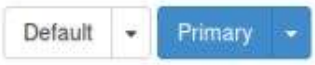
```
<div >
   <button type="button"
      data-toggle="dropdown">
      Default <span ></span>
   </button>
   <ul >
      <li><a href="#">Action</a></li>
      <li><a href="#">Another action</a></li>
      <li><a href="#">Something else here</a></li>
      <li ></li>
      <li><a href="#">Separated link</a></li>
   </ul>
</div>
```

```
<div >
   <button type="button"
      data-toggle="dropdown">
      Primary <span ></span>
   </button>
   <ul >
      <li><a href="#">Action</a></li>
      <li><a href="#">Another action</a></li>
      <li><a href="#">Something else here</a></li>
      <li ></li>
      <li><a href="#">Separated link</a></li>
   </ul>
</div>
```

Split Button Dropdowns

Split button dropdowns use the same general style as the dropdown button but add a primary action along with the dropdown. Split buttons have the primary action on the left and a toggle on the right that displays the dropdown.

```
<div >
   <button type="button" >Default</button>
   <button type="button"
      data-toggle="dropdown">
      <span ></span>
      <span >Toggle Dropdown</span>
   </button>
   <ul >
      <li><a href="#">Action</a></li>
      <li><a href="#">Another action</a></li>
      <li><a href="#">Something else here</a></li>
      <li ></li>
      <li><a href="#">Separated link</a></li>
   </ul>
</div>
<div >
   <button type="button" >Primary</button>
   <button type="button"
      data-toggle="dropdown">
      <span ></span>
      <span >Toggle Dropdown</span>
   </button>
   <ul >
      <li><a href="#">Action</a></li>
      <li><a href="#">Another action</a></li>
      <li><a href="#">Something else here</a></li>
      <li ></li>
      <li><a href="#">Separated link</a></li>
   </ul>
</div>
```



Button Dropdown Size

You can use the dropdowns with any button size: **.btn-large, .btn-sm,** or **.btn-xs**.

```
<div >
   <button type="button"
      data-toggle="dropdown">
      Default <span ></span>
   </button>
   <ul >
      <li><a href="#">Action</a></li>
      <li><a href="#">Another action</a></li>
```

```
      <li><a href="#">Something else here</a></li>
      <li ></li>
      <li><a href="#">Separated link</a></li>
   </ul>
</div>
<div >
   <button type="button"
      data-toggle="dropdown">
      Primary <span ></span>
   </button>
   <ul >
      <li><a href="#">Action</a></li>
      <li><a href="#">Another action</a></li>
      <li><a href="#">Something else here</a></li>
      <li ></li>
      <li><a href="#">Separated link</a></li>
   </ul>
</div>
<div >
   <button type="button"
      data-toggle="dropdown">
      Success <span ></span>
   </button>
   <ul >
      <li><a href="#">Action</a></li>
      <li><a href="#">Another action</a></li>
      <li><a href="#">Something else here</a></li>
      <li ></li>
      <li><a href="#">Separated link</a></li>
   </ul>
</div>
```

Dropup Variation

Menus can also be built to drop up rather than down. To achieve this, simply add **.dropup** to the parent **.btn-group** container.

```
<div >
   <div >
      <button type="button"
         data-toggle="dropdown">
         Default <span ></span>
      </button>
      <ul >
         <li><a href="#">Action</a></li>
         <li><a href="#">Another action</a></li>
         <li><a href="#">Something else here</a></li>
         <li ></li>
         <li><a href="#">Separated link</a></li>
      </ul>
   </div>
   <div >
      <button type="button"
         data-toggle="dropdown">
         Primary <span ></span>
      </button>
      <ul >
         <li><a href="#">Action</a></li>
         <li><a href="#">Another action</a></li>
         <li><a href="#">Something else here</a></li>
         <li ></li>
         <li><a href="#">Separated link</a></li>
      </ul>
   </div>
</div>
```

BOOTSTRAP INPUT GROUPS

This chapter explains about one more feature Bootstrap supports, the Input Groups. Input groups are extended Form Controls. Using input groups you can easily prepend and append text or buttons to the text-based inputs.

By adding prepended and appended content to an input field, you can add common elements to the user's input. For example, you can add the dollar symbol, the @ for a Twitter username, or anything else that might be common for your application interface.

To prepend or append elements to a **.form-control**:

- Wrap it in a with class **.input-group**
- As a next step, within that same , place your extra content inside a with class **.input-group-addon**.
- Now place this either before or after the <input> element.

> *For cross browser compatibility, avoid using <select> elements here as they cannot be fully styled in WebKit browsers. Also do not apply input group classes directly to form groups. An input group is an isolated component.*

Basic Input Group

The following example demonstrates basic input group:

```
<div style="padding: 100px 100px 10px;">
   <form >
      <div >
         <span >@</span>
         <input type="text" >
      </div>
      <br>

      <div >
         <input type="text" >
         <span >.00</span>
      </div>
      <br>
      <div >
         <span >$</span>
         <input type="text" >
         <span >.00</span>
      </div>
   </form>
</div>
```

Input Group Sizing

You can change the size of the input groups, by adding the relative form sizing classes like **.input-group-lg, input-group-sm, input-group-xs** to the **.input-group** itself. The contents within will

automatically resize.

Following examples demonstrates this:

```
<div style="padding: 100px 100px 10px;">
   <form >
      <div >
         <span >@</span>
         <input type="text" >
      </div><br>

      <div >
         <span >@</span>
         <input type="text" >
      </div><br>

      <div >
         <span >@</span>
         <input type="text" >
      </div>
   </form>
</div>
```

Checkboxes and Radio Addons

You can preappend or append radio buttons and checkboxes instead of text as demonstrated in the following example:

```
<div style="padding: 100px 100px 10px;">
   <form >
      <div >
         <div >
            <div >
               <span >
                  <input type="checkbox">
               </span>
               <input type="text" >
            </div><!-- /input-group -->
         </div><!-- /.col-lg-6 --><br>
         <div >
            <div >
               <span >
                  <input type="radio">
               </span>
               <input type="text" >
            </div><!-- /input-group -->
         </div><!-- /.col-lg-6 -->
      </div><!-- /.row -->
   </form>
</div>
```

Button Addons

You can even preappend or append buttons in input groups. Instead of **.input-group-addon**

class, you'll need to use class **.input-group-btn** to wrap the buttons. This is required due to the default browser styles that cannot be overridden. Following examples demonstrates this:

```
<div style="padding: 100px 100px 10px;">
   <form >
      <div >
         <div >
            <div >
               <span >
                  <button >
                     Go!
                  </button>
               </span>
               <input type="text" >
            </div><!-- /input-group -->
         </div><!-- /.col-lg-6 --><br>
         <div >
            <div >
               <input type="text" >
               <span >
                  <button >
                     Go!
                  </button>
               </span>
            </div><!-- /input-group -->
         </div><!-- /.col-lg-6 -->
      </div><!-- /.row -->
   </form>
</div>
```

Buttons with Dropdowns

Adding buttons with dropdown menus in input groups can be done by simply wrapping the button and dropdown menu in a **.input-group-btn** class as demonstrated in the following example:

```
<div style="padding: 100px 100px 10px;">
   <form >
      <div >
         <div >
            <div >
               <div >
                  <button type="button" class="btn btn-default
                     dropdown-toggle" data-toggle="dropdown">
                     DropdownMenu
                     <span ></span>
                  </button>
                  <ul >
                     <li><a href="#">Action</a></li>
                     <li><a href="#">Another action</a></li>
                     <li><a href="#">Something else here</a></li>
                     <li ></li>
                     <li><a href="#">Separated link</a></li>
                  </ul>
               </div><!-- /btn-group -->
               <input type="text" >
            </div><!-- /input-group -->
         </div><!-- /.col-lg-6 --><br>
         <div >
            <div >
               <input type="text" >
               <div >
                  <button type="button" class="btn btn-default
                     dropdown-toggle" data-toggle="dropdown">
                     DropdownMenu
                     <span ></span>
                  </button>
```

```
                  <ul >
                     <li><a href="#">Action</a></li>
                     <li><a href="#">Another action</a></li>
                     <li><a href="#">Something else here</a></li>
                     <li ></li>
                     <li><a href="#">Separated link</a></li>
                  </ul>
               </div><!-- /btn-group -->
            </div><!-- /input-group -->
         </div><!-- /.col-lg-6 -->
      </div><!-- /.row -->
   </form>
</div>
```

Segmented Buttons

To segment button dropdowns in input groups, use the same general style as the dropdown button, but add a primary action along with the dropdown as can be seen in the following example:

```
<div style="padding: 100px 100px 10px;">
   <form >
      <div >
         <div >
            <div >
               <div >
                  <button type="button"
                     tabindex="-1">Dropdwon Menu
                  </button>
                  <button type="button" class="btn btn-default
                     dropdown-toggle" data-toggle="dropdown" tabindex="-1">
                     <span ></span>
                     <span >Toggle Dropdown</span>
                  </button>
                  <ul >
                     <li><a href="#">Action</a></li>
                     <li><a href="#">Another action</a></li>
                     <li><a href="#">Something else here</a></li>
                     <li ></li>
                     <li><a href="#">Separated link</a></li>
                  </ul>
               </div><!-- /btn-group -->
               <input type="text" >
            </div><!-- /input-group -->
         </div><!-- /.col-lg-6 --><br>
         <div >
            <div >
               <input type="text" >
               <div >
                  <button type="button"
                     tabindex="-1">Dropdwon Menu
                  </button>
                  <button type="button" class="btn btn-default
                     dropdown-toggle" data-toggle="dropdown" tabindex="-1">
                     <span ></span>
                     <span >Toggle Dropdown</span>
                  </button>
                  <ul >
                     <li><a href="#">Action</a></li>
                     <li><a href="#">Another action</a></li>
                     <li><a href="#">Something else here</a></li>
                     <li ></li>
                     <li><a href="#">Separated link</a></li>
                  </ul>
               </div><!-- /btn-group -->
            </div><!-- /input-group -->
         </div><!-- /.col-lg-6 -->
```

```
      </div><!-- /.row -->
   </form>
</div>
```

BOOTSTRAP NAVIGATION ELEMENTS

Bootstrap provides a few different options for styling navigation elements. All of them share the same markup and base class, **.nav**. Bootstrap also provides a helper class, to share markup and states. Swap modifier classes to switch between each style.

Tabular Navigation or Tabs

To create a tabbed navigation menu:

- Start with a basic unordered list with the base class of **.nav**
- Add class **.nav-tabs**.

The following example demonstrates this:

```
<p>Tabs Example</p>
<ul >
   <li >Home</a></li>
   <li><a href="#">SVN</a></li>
   <li><a href="#">iOS</a></li>
   <li><a href="#">VB.Net</a></li>
   <li><a href="#">Java</a></li>
   <li><a href="#">PHP</a></li>
</ul>
```

Pills Navigation

Basic Pills

To turn the tabs into pills, follow the same steps as above, use the class **.nav-pills** instead of **.nav-tabs**.

The following example demonstrates this:

```
<p>Pills Example</p>
<ul >
   <li >Home</a></li>
   <li><a href="#">SVN</a></li>
   <li><a href="#">iOS</a></li>
   <li><a href="#">VB.Net</a></li>
   <li><a href="#">Java</a></li>
   <li><a href="#">PHP</a></li>
</ul>
```

Verticle Pills

You can stack the pills vertically using the class **.nav-stacked** along with the classes: **.nav, .nav-pills**.

The following example demonstrates this:

```
<p>Vertical Pills Example</p>
<ul >
   <li >Home</a></li>
   <li><a href="#">SVN</a></li>
   <li><a href="#">iOS</a></li>
   <li><a href="#">VB.Net</a></li>
   <li><a href="#">Java</a></li>
   <li><a href="#">PHP</a></li>
</ul>
```

Justified Nav

You can make tabs or pills of equal widths as of their parent at screens wider than 768px using class **.nav-justified** along with **.nav, .nav-tabs** or **.nav, .nav-pills** respectively. On smaller screens, the nav links are stacked.

The following example demonstrates this:

```
<p>Justified Nav Elements Example</p>
<ul >
   <li >Home</a></li>
   <li><a href="#">SVN</a></li>
   <li><a href="#">iOS</a></li>
   <li><a href="#">VB.Net</a></li>
   <li><a href="#">Java</a></li>
   <li><a href="#">PHP</a></li>
</ul><br><br><br>

<ul >
   <li >Home</a></li>
   <li><a href="#">SVN</a></li>
   <li><a href="#">iOS</a></li>
   <li><a href="#">VB.Net</a></li>
   <li><a href="#">Java</a></li>
   <li><a href="#">PHP</a></li>
</ul>
```

Disabled Links

For each of the **.nav** classes, if you add the **.disabled** class, it will create a gray link that also disables the **:hover** state as shown in the following example:

```
<p>Disabled Link Example</p>
<ul >
   <li >Home</a></li>
   <li><a href="#">SVN</a></li>
   <li >iOS(disabled link)</a></li>
   <li><a href="#">VB.Net</a></li>
   <li><a href="#">Java</a></li>
   <li><a href="#">PHP</a></li>
</ul><br><br>

<ul >
   <li >Home</a></li>
   <li><a href="#">SVN</a></li>
   <li><a href="#">iOS</a></li>
   <li  >VB.Net(disabled link)</a></li>
   <li><a href="#">Java</a></li>
   <li><a href="#">PHP</a></li>
</ul>
```

> *This class will only change the <a>'s appearance, not its functionality. Use custom JavaScript to disable links here.*

Dropdowns

Navigation menus share a similar syntax with dropdown menus. By default, you have a list item that has an anchor working in conjunction with some data-attributes to trigger an unordered list with a **.dropdown-menu** class.

Tabs with Dropdowns

To add dropdowns to tab:

- Start with a basic unordered list with the base class of **.nav**
- Add the class **.nav-tabs**.
- Now add an unordered list with a **.dropdown-menu** class.

```
<p>Tabs With Dropdown Example</p>
<ul >
   <li >Home</a></li>
   <li><a href="#">SVN</a></li>
   <li><a href="#">iOS</a></li>
   <li><a href="#">VB.Net</a></li>
   <li >
      <a >
         Java <span ></span>
      </a>
      <ul >
         <li><a href="#">Swing</a></li>
         <li><a href="#">jMeter</a></li>
```

```
         <li><a href="#">EJB</a></li>
         <li ></li>
         <li><a href="#">Separated link</a></li>
      </ul>
   </li>
   <li><a href="#">PHP</a></li>
</ul>
```

Pills with Dropdowns

To do the same thing with pills, simply swap the **.nav-tabs** class with **.nav-pills** as shown in the following example.

```
<p>Pills With Dropdown Example</p>
<ul >
   <li >Home</a></li>
   <li><a href="#">SVN</a></li>
   <li><a href="#">iOS</a></li>
   <li><a href="#">VB.Net</a></li>
   <li >
      <a >
         Java <span ></span>
      </a>
      <ul >
         <li><a href="#">Swing</a></li>
         <li><a href="#">jMeter</a></li>
         <li><a href="#">EJB</a></li>
         <li ></li>
         <li><a href="#">Separated link</a></li>
      </ul>
   </li>
   <li><a href="#">PHP</a></li>
</ul>
```

BOOTSTRAP NAVBAR

The navbar is one of the prominent features of Bootstrap sites. Navbars are responsive 'meta' components that serve as navigation headers for your application or site. Navbars collapse in mobile views and become horizontal as the available viewport width increases. At its core, the navbar includes styling for site names and basic navigation.

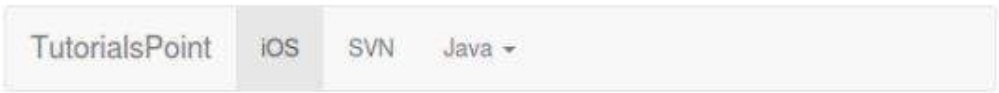
Default Navbar

To create a default navbar:

- Add the classes **.navbar, .navbar-default** to the <nav> tag.
- Add **role="navigation"** to the above element, to help with accessibility.
- Add a header class **.navbar-header** to the element. Include an <a> element with class **navbar-brand**. This will give the text a slightly larger size.
- To add links to the navbar, simply add an unordered list with the classes of **.nav, .navbar-nav**.

The following example demonstrates this:

```
<nav >
   <div >
      <a >TutorialsPoint</a>
   </div>
   <div>
      <ul >
         <li >iOS</a></li>
         <li><a href="#">SVN</a></li>
         <li >
            <a href="#" >
               Java
               <b ></b>
            </a>
            <ul >
               <li><a href="#">jmeter</a></li>
               <li><a href="#">EJB</a></li>
               <li><a href="#">Jasper Report</a></li>
               <li ></li>
               <li><a href="#">Separated link</a></li>
               <li ></li>
               <li><a href="#">One more separated link</a></li>
            </ul>
         </li>
      </ul>
   </div>
</nav>
```



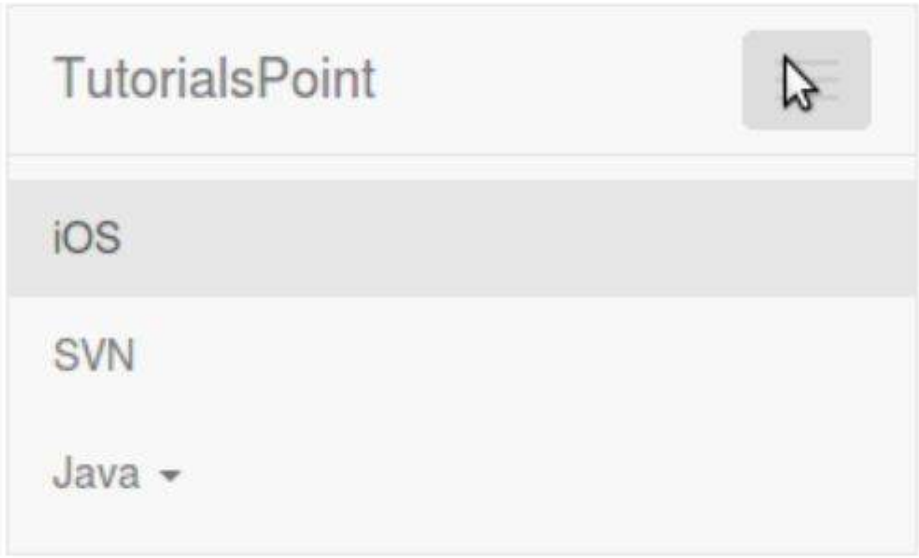
Responsive Navbar

To add responsive features to the navbar, the content that you want to be collapsed needs to be wrapped in a with classes **.collapse, .navbar-collapse**. The collapsing nature is tripped by a button that has the class of **.navbar-toggle** and then features two data- elements. The first, **data-toggle**, is used to tell the JavaScript what to do with the button, and the second, **data-target**, indicates which element to toggle. Then with a class **.icon-bar** create what we like to call the hamburger button. This will toggle the elements that are in the **.nav-collapse** . For this feature to work, you need to include the Bootstrap Collapse Plugin.

The following example demonstrates this:

```
<nav >
   <div >
      <button type="button"
         data-target="#example-navbar-collapse">
         <span >Toggle navigation</span>
         <span ></span>
         <span ></span>
         <span ></span>
      </button>
      <a >TutorialsPoint</a>
   </div>
   <div >
      <ul >
```

```
         <li >iOS</a></li>
         <li><a href="#">SVN</a></li>
         <li >
            <a href="#" >
               Java <b ></b>
            </a>
            <ul >
               <li><a href="#">jmeter</a></li>
               <li><a href="#">EJB</a></li>
               <li><a href="#">Jasper Report</a></li>
               <li ></li>
               <li><a href="#">Separated link</a></li>
               <li ></li>
               <li><a href="#">One more separated link</a></li>
            </ul>
         </li>
      </ul>
   </div>
</nav>
```

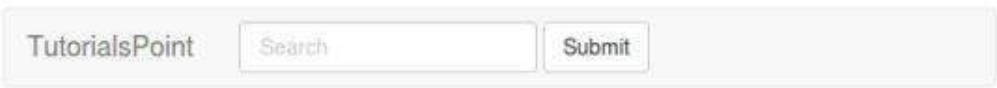


Forms in Navbar

Instead of using the default class-based forms from Chapter Bootstrap Forms, forms that are in the navbar, use the **.navbar-form** class. This ensures that the form's proper vertical alignment and collapsed behavior in narrow viewports. Use the alignment options *explainedinComponentalignmentsection* to decide where it resides within the navbar content.

The following example demonstrates this:

```
<nav >
   <div >
      <a >TutorialsPoint</a>
   </div>
   <div>
      <form >
         <div >
            <input type="text" >
         </div>
         <button type="submit" >Submit</button>
      </form>
   </div>
</nav>
```



Buttons in Navbar

You can add buttons using class **.navbar-btn** to <button> elements not residing in a <form> to vertically center them in the navbar. **.navbar-btn** can be used on <a> and <input> elements.

> *Do not use **.navbar-btn** nor the standard button classes on <a> elements within **.navbar-nav**.*

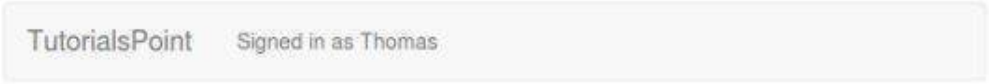
The following example demonstrates this:

```
<nav >
   <div >
      <a >TutorialsPoint</a>
   </div>
   <div>
      <form >
         <div >
            <input type="text" >
         </div>
         <button type="submit" >Submit Button</button>
      </form>
      <button type="button" >
         Navbar Button
      </button>
   </div>
</nav>
```

Text in Navbar

To wrap strings of text in an element use the class **.navbar-text**. This is usually used with <p> tag for proper leading and color. The following example demonstrates this:

```
<nav >
   <div >
      <a >TutorialsPoint</a>
   </div>
   <div>
      <p >Signed in as Thomas</p>
   </div>
</nav>
```

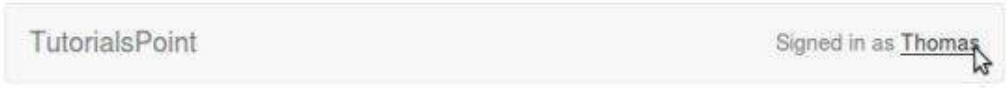


Non-nav Links

If you want to use the standard links that are not within the regular navbar navigation component, then use the class **navbar-link** to add proper colors for the default and inverse navbar options as shown in the following example:

```
<nav >
   <div >
      <a >TutorialsPoint</a>
   </div>
   <div>
      <p >Signed in as
         <a href="#" >Thomas</a>
      </p>
   </div>
```

```
</div>
```



Component Alignment

You can align the components like *nav links, forms, buttons, or text* to left or right in a navbar using the utility classes **.navbar-left** or **.navbar-right**. Both classes will add a CSS float in the specified direction. The following example demonstrates this:

```
<nav >
   <div >
      <a >TutorialsPoint</a>
   </div>
   <div>
      <!--Left Align-->
      <ul >
         <li >
            <a href="#" >
               Java
               <b ></b>
            </a>
            <ul >
               <li><a href="#">jmeter</a></li>
               <li><a href="#">EJB</a></li>
               <li><a href="#">Jasper Report</a></li>
               <li ></li>
               <li><a href="#">Separated link</a></li>
               <li ></li>
               <li><a href="#">One more separated link</a></li>
            </ul>
         </li>
      </ul>
      <form >
         <button type="submit" >
            Left align-Submit Button
         </button>
      </form>
      <p >Left align-Text</p>
      <!--Right Align-->
      <ul >
         <li >
            <a href="#" >
               Java <b ></b>
            </a>
            <ul >
               <li><a href="#">jmeter</a></li>
               <li><a href="#">EJB</a></li>
               <li><a href="#">Jasper Report</a></li>
               <li ></li>
               <li><a href="#">Separated link</a></li>
               <li ></li>
               <li><a href="#">One more separated link</a></li>
            </ul>
         </li>
      </ul>
      <form >
         <button type="submit" >
            Right align-Submit Button
         </button>
      </form>
      <p >Right align-Text</p>
   </div>
</nav>
```

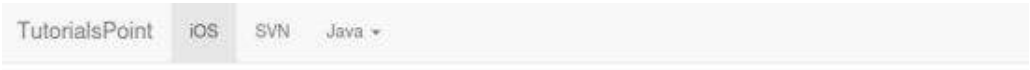
Fixed to Top

The Bootstrap navbar can be dynamic in its positioning. By default, it is a block-level element that takes its positioning based on its placement in the HTML. With a few helper classes, you can place it either on the top or bottom of the page, or you can make it scroll statically with the page.

If you want the navbar fixed to the top, add class **.navbar-fixed-top** to the **.navbar class**. The following example demonstrates this:

> *To prevent the navbar from sitting on top of other content in the body of the page, add at least 50 pixels of padding to the <body> tag or try your own values.*

```
<nav >
   <div >
      <a >TutorialsPoint</a>
   </div>
   <div>
      <ul >
         <li >iOS</a></li>
         <li><a href="#">SVN</a></li>
         <li >
            <a href="#" >
               Java <b ></b>
            </a>
            <ul >
               <li><a href="#">jmeter</a></li>
               <li><a href="#">EJB</a></li>
               <li><a href="#">Jasper Report</a></li>
               <li ></li>
               <li><a href="#">Separated link</a></li>
               <li ></li>
               <li><a href="#">One more separated link</a></li>
            </ul>
         </li>
      </ul>
   </div>
</nav>
```



Fixed to Bottom

If you want the navbar fixed to the bottom of the page, add class **.navbar-fixed-bottom** to the **.navbar class**. The following example demonstrates this:

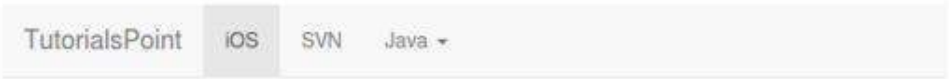
```
<nav >
   <div >
      <a >TutorialsPoint</a>
   </div>
   <div>
      <ul >
         <li >iOS</a></li>
         <li><a href="#">SVN</a></li>
         <li >
            <a href="#" >
               Java <b ></b>
            </a>
            <ul >
               <li><a href="#">jmeter</a></li>
               <li><a href="#">EJB</a></li>
               <li><a href="#">Jasper Report</a></li>
               <li ></li>
               <li><a href="#">Separated link</a></li>
               <li ></li>
```

```
               <li><a href="#">One more separated link</a></li>
            </ul>
         </li>
      </ul>
   </div>
</nav>
```

Static Top

To create a navbar that scrolls with the page, add the **.navbar-static-top** class. This class does not require adding the padding to the <body>.

```
<nav >
   <div >
      <a >TutorialsPoint</a>
   </div>
   <div>
      <ul >
         <li >iOS</a></li>
         <li><a href="#">SVN</a></li>
         <li >
            <a href="#" >
               Java <b ></b>
            </a>
            <ul >
               <li><a href="#">jmeter</a></li>
               <li><a href="#">EJB</a></li>
               <li><a href="#">Jasper Report</a></li>
               <li ></li>
               <li><a href="#">Separated link</a></li>
               <li ></li>
               <li><a href="#">One more separated link</a></li>
            </ul>
         </li>
      </ul>
   </div>
</nav>
```



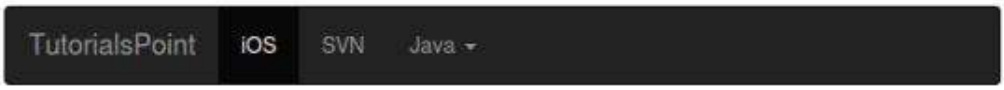
Inverted Navbar

To create an inverted navbar with a black background and with white text, simply add the **.navbar-inverse** class to the **.navbar** class as demonstrated in the following example:

> *To prevent the navbar from sitting on top of other content in the body of the page, add at least 50 pixels of padding to the <body> tag or try your own values.*

```
<nav >
   <div >
      <a >TutorialsPoint</a>
   </div>
   <div>
```

```
      <ul >
         <li >iOS</a></li>
         <li><a href="#">SVN</a></li>
         <li >
            <a href="#" >
               Java <b ></b>
            </a>
            <ul >
               <li><a href="#">jmeter</a></li>
               <li><a href="#">EJB</a></li>
               <li><a href="#">Jasper Report</a></li>
               <li ></li>
               <li><a href="#">Separated link</a></li>
               <li ></li>
               <li><a href="#">One more separated link</a></li>
            </ul>
         </li>
      </ul>
   </div>
</nav>
```



BOOTSTRAP BREADCRUMB

Breadcrumbs are a great way to show hierarchy-based information for a site. In the case of blogs, breadcrumbs can show the dates of publishing, categories, or tags. They indicate the current page's location within a navigational hierarchy.

A breadcrumb in Bootstrap is simply an unordered list with a class of **.breadcrumb**. The separator is automatically added by CSS *bootstrap. min. css* through the following class:

```
.breadcrumb > li + li:before {
    color: #CCCCCC;
    content: "/ ";
    padding: 0 5px;
}
```

The following example demonstrates breadcrumbs:

```
<ol >
  <li><a href="#">Home</a></li>
  <li><a href="#">2013</a></li>
  <li >November</li>
</ol>
```

BOOTSTRAP PAGINATION

This chapter discusses about the pagination feature that Bootstrap supports. Pagination, an unordered list is handled by Bootstrap like a lot of other interface elements.

Pagination

The following table lists the classes that Bootstrap provides to handle pagination.

Class	Description	Sample code
.pagination	Add this class to get the pagination on your page.	`<ul class="pagination"> <li><a href="#">&laquo;</a></li> <li><a`

		`href="#">1</a></li>`
		`  .....`
		`</ul>`

.disabled, .active	You can customize links by using **.disabled** for unclickable links and **.active** to indicate the current page.	`<ul class="pagination">`
		`  <li >&laquo;</a></li>`
		`  <li >(current)</span></a></li>`
		`  .....`
		`</ul>`

.pagination-lg, .pagination-sm	Use these classes to get different size items.	`<ul class="pagination pagination-lg">...</ul>`
		`<ul >...</ul>`
		`<ul >...</ul>`

Default Pagination

The following example demonstrates the use of class **.pagination** discussed in the above table:

```
<ul >
  <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>
```

« 1 2 3 4 5 »

States

The following example demonstrates the use of class **.disabled, .active** discussed in the above table:

```
<ul >
  <li><a href="#">&laquo;</a></li>
  <li >1</a></li>
  <li >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>
```

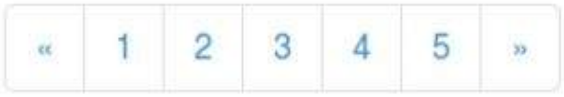
« 1 2 3 4 5 »

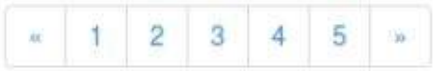
Sizing

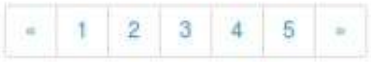
The following example demonstrates the use of classes for sizing, **.pagination-*** discussed in the above table:

```
<ul >
  <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><br>
<ul >
  <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><br>
<ul >
 <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>
```







Pager

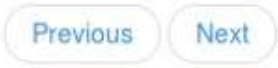
If you need to create simple pagination links that go beyond text, the pager can work quite well. Like the pagination links, the pager is an unordered list. By default the links are centered. The following table lists the classes Bootstrap provides for pager.

Class	Description	Sample code
.pager	Add this class to get the pager links.	`<ul class="pager"> <li><a href="#">Previous</a></li> <li><a href="#">Next</a></li> </ul>`
.previous, .next	Use class **.previous** to left align and **.next** to right-align the links.	`<ul class="pager"> <li >&larr; Older</a></li> <li >Newer &rarr;</a></li> </ul>`
.disabled	Add this class to get a muted look.	`<ul class="pager"> <li >&larr; Older</a></li> <li >Newer &rarr;</a></li> </ul>`

Default Pager

The following example demonstrates the use of class **.pager** discussed in the above table:

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



Aligned Links

The following example demonstrates the use of classes for alignment, **.previous, .next** discussed in the above table:

```
<ul >
  <li >&larr; Older</a></li>
  <li >Newer &rarr;</a></li>
</ul>
```

States

The following example demonstrates the use of class **.disabled** discussed in the above table:

```
<ul >
  <li >&larr; Older</a></li>
  <li >Newer &rarr;</a></li>
</ul>
```

BOOTSTRAP LABELS

This chapter covers Bootstrap labels. Labels are great for offering counts, tips, or other markup for pages. Use class **.label** to display labels as shown in the following example:

```
<h1>Example Heading <span >Label</span></h1>
<h2>Example Heading <span >Label</span></h2>
<h3>Example Heading <span >Label</span></h3>
<h4>Example Heading <span >Label</span></h4>
```

You can modify the appearance of the labels using the modifier classes such as, **label-default, label-primary, label-success, label-info, label-warning, label-danger** as shown in the following example:

```
<span >Default Label</span>
<span >Primary Label</span>
<span >Success Label</span>
<span >Info Label</span>
<span >Warning Label</span>
<span >Danger Label</span>
```

BOOTSTRAP BADGES

This chapter will discuss about Bootstrap badges. Badges are similar to labels; the primary difference is that the corners are more rounded.

Badges are mainly used to highlight new or unread items. To use badges just add **** to links, Bootstrap navs, and more.

The following example demonstrates this:

```
<a href="#">Mailbox <span >50</span></a>
```

When there are no new or unread items, badges will simply collapse via CSS's **:empty** selector, provided no content exists within.

Active Nav States

You can place badges in active states of pill and list navigations. You can achieve this by placing **** to active links, as demonstrated in the following example:

```
<h4>Example for Active State in Pill </h4>
<ul >
   <li >42</span></a></li>
   <li><a href="#">Profile</a></li>
   <li><a href="#">Messages <span >3</span></a></li>
</ul>
<br>
<h4>Example for Active State in navigations</h4>
<ul >
   <li >
      <a href="#">
         <span >42</span>
         Home
      </a>
   </li>
   <li><a href="#">Profile</a></li>
   <li>
      <a href="#">
         <span >3</span>
         Messages
      </a>
   </li>
</ul>
```

BOOTSTRAP JUMBOTRON

This chapter will discuss one more feature that Bootstrap supports, the Jumbotron. As the name suggest this component can optionally increase the size of headings and add a lot of margin for landing page content. To use the Jumbotron:

- Create a container with the class of **.jumbotron**.
- In addition to a larger <h1>, the *font-weight* is reduced to 200px.

The following example demonstrates this:

```
<div >
   <div >
      <h1>Welcome to landing page!</h1>
      <p>This is an example for jumbotron.</p>
      <p><a >
         Learn more</a>
      </p>
   </div>
</div>
```

To get a jumbotron of full width, and without rounded corners use the **.jumbotron** class outside all **.container** classes and instead add a **.container** within, as shown in the following example:

```
<div >
   <div >
      <h1>Welcome to landing page!</h1>
      <p>This is an example for jumbotron.</p>
      <p><a >
         Learn more</a>
      </p>
   </div>
</div>
```

The page header is a nice little feature to add appropriate spacing around the headings on a page. This is particularly helpful on a web page where you may have several post titles and need a way to add distinction to each of them. To use a page header, wrap your heading in a with a class of **.page-header**:

```
<div >
   <h1>Example page header
      <small>Subtext for header</small>
   </h1>
</div>
<p>This is a sample text.This is a sample text.This is a sample text.
   This is a sample text.</p>
```

Example page header Subtext for header

This is a sample text.This is a sample text.This is a sample text.This is a sample text.

BOOTSTRAP THUMBNAILS

This chapter discusses about Bootstrap thumbnails. A lot of sites need a way to lay out images, videos, text, etc, in a grid, and Bootstrap has an easy way to do this with thumbnails. To create thumbnails using Bootstrap:

- Add an <a> tag with the class of **.thumbnail** around an image.
- This adds four pixels of padding and a gray border.
- On hover, an animated glow outlines the image.

The following example demonstrates a default thumbnail:

```
<div >
   <div >
      <a href="#" >
         <img src="/bootstrap/images/kittens.jpg"
         alt="Generic placeholder thumbnail">
      </a>
   </div>
   <div >
      <a href="#" >
         <img src="/bootstrap/images/kittens.jpg"
         alt="Generic placeholder thumbnail">
      </a>
   </div>
   <div >
      <a href="#" >
         <img src="/bootstrap/images/kittens.jpg"
         alt="Generic placeholder thumbnail">
      </a>
   </div>
   <div >
      <a href="#" >
         <img src="/bootstrap/images/kittens.jpg"
         alt="Generic placeholder thumbnail">
      </a>
   </div>
</div>
```

Adding Custom Content

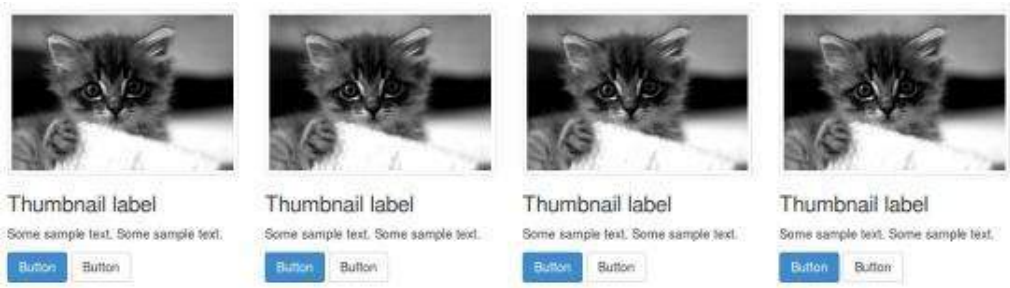
Now that we have a basic thumbnail, it's possible to add any kind of HTML content like headings, paragraphs, or buttons into thumbnails. Follow the steps below:

- Change the <a> tag that has a class of **.thumbnail** to a <div>.
- Inside of that <div>, you can add anything you need. As this is a <div>, we can use the default span-based naming convention for sizing.
- If you want to group multiple images, place them in an unordered list, and each list item will be floated to the left.

The following example demonstrates this:

```
<div >
   <div >
      <div >
         <img src="/bootstrap/images/kittens.jpg"
         alt="Generic placeholder thumbnail">
      </div>
      <div >
         <h3>Thumbnail label</h3>
         <p>Some sample text. Some sample text.</p>
         <p>
            <a href="#" >
               Button
            </a>
            <a href="#" >
               Button
            </a>
         </p>
      </div>
   </div>
   <div >
      <div >
         <img src="/bootstrap/images/kittens.jpg"
         alt="Generic placeholder thumbnail">
      </div>
      <div >
         <h3>Thumbnail label</h3>
         <p>Some sample text. Some sample text.</p>
         <p>
            <a href="#" >
               Button
            </a>
            <a href="#" >
               Button
            </a>
         </p>
      </div>
   </div>
   <div >
      <div >
         <img src="/bootstrap/images/kittens.jpg"
         alt="Generic placeholder thumbnail">
      </div>
      <div >
         <h3>Thumbnail label</h3>
         <p>Some sample text. Some sample text.</p>
         <p>
            <a href="#" >
               Button
            </a>
            <a href="#" >
               Button
            </a>
         </p>
      </div>
   </div>
   <div >
      <div >
         <img src="/bootstrap/images/kittens.jpg"
         alt="Generic placeholder thumbnail">
      </div>
      <div >
```

```
         <h3>Thumbnail label</h3>
         <p>Some sample text. Some sample text.</p>
         <p>
            <a href="#" >
               Button
            </a>
            <a href="#" >
               Button
            </a>
         </p>
      </div>
   </div>
</div>
```



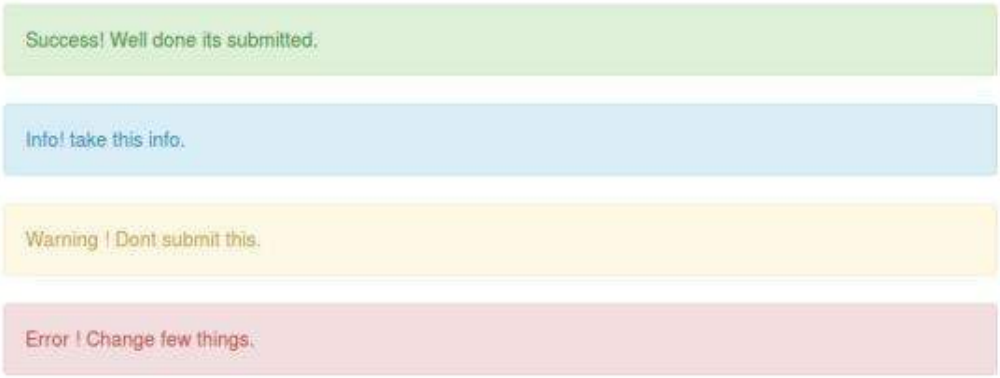
BOOTSTRAP ALERTS

This chapter will discuss about alerts and the classes Bootstrap provides for alerts. Alerts provide a way to style messages to the user. They provide contextual feedback messages for typical user actions.

You can add an optional close icon to alert. For inline dismissal use the Alerts jQuery plugin.

You can add a basic alert by creating a wrapper and adding a class of **.alert** and one of the four contextual classes (e.g., **.alert-success, .alert-info, .alert-warning, .alert-danger**). The following example demonstrates this:

```
<div >Success! Well done its submitted.</div>
<div >Info! take this info.</div>
<div >Warning ! Dont submit this.</div>
<div >Error ! Change few things.</div>
```



Dismissal Alerts

To build a dismissal alert:

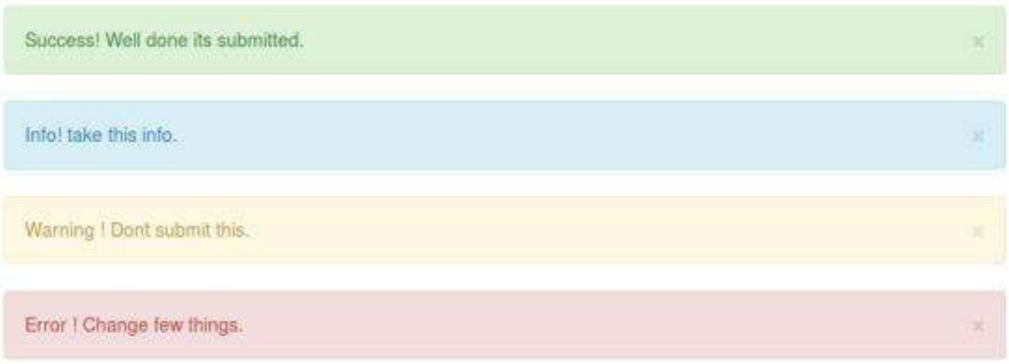
- Add a basic alert by creating a wrapper and adding a class of **.alert** and one of the four contextual classes (e.g., **.alert-success, .alert-info, .alert-warning, .alert-danger**)
- Also add optional **.alert-dismissable** to the above class.
- Add a close button.

The following example demonstrates this:

```
<div >
   <button type="button"
```

```
      aria-hidden="true">
      &times;
   </button>
   Success! Well done its submitted.
</div>
<div >
   <button type="button"
      aria-hidden="true">
      &times;
   </button>
   Info! take this info.
</div>
<div >
   <button type="button"
      aria-hidden="true">
      &times;
   </button>
   Warning ! Dont submit this.
</div>
<div >
   <button type="button"
      aria-hidden="true">
      &times;
   </button>
   Error ! Change few things.
</div>
```

> *Be sure to use the <button> element with the data-dismiss="alert" data attribute.*

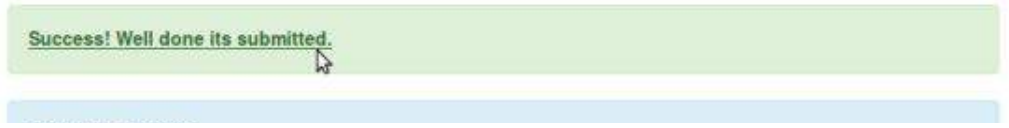


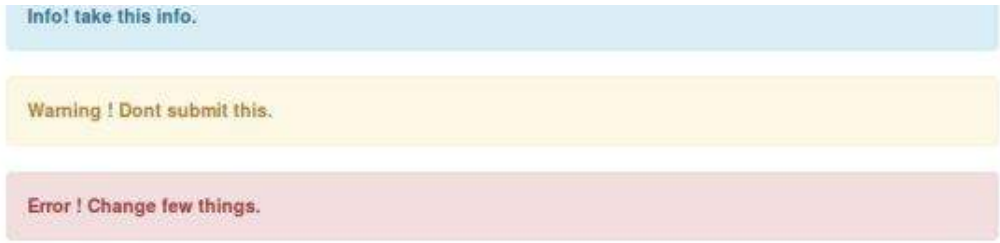
Links in Alerts

To get links in alerts:

- Add a basic alert by creating a wrapper and adding a class of **.alert** and one of the four contextual classes (e.g., **.alert-success, .alert-info, .alert-warning, .alert-danger**)
- Use the **.alert-link** utility class to quickly provide matching colored links within any alert.

```
<div >
   <a href="#" >Success! Well done its submitted.</a>
</div>
<div >
   <a href="#" >Info! take this info.</a>
</div>
<div >
   <a href="#" >Warning ! Dont submit this.</a>
</div>
<div >
   <a href="#" >Error ! Change few things.</a>
</div>
```





BOOTSTRAP PROGRESS BARS

This chapter discusses about Bootstrap progress bars. The purpose of progress bars is to show that assets are loading, in progress, or that there is action taking place regarding elements on the page.

> *Progress bars use CSS3 transitions and animations to achieve some of their effects. These features are not supported in Internet Explorer 9 and below or older versions of Firefox. Opera 12 does not support animations.*

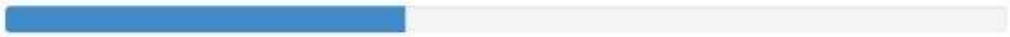
Default Progress Bar

To create a basic progress bar:

- Add a with a class of **.progress**.
- Next, inside the above , add an empty with a class of **.progress-bar**.
- Add a style attribute with the width expressed as a percentage. Say for example, style="60%"; indicates that the progress bar was at 60%.

Let us see an example below:

```
<div >
   <div
      aria-valuemin="0" aria-valuemax="100" style="width: 40%;">
      <span >40% Complete</span>
   </div>
</div>
```



Alternate Progress Bar

To create a progress bar with different styles:

- Add a with a class of **.progress**.
- Next, inside the above , add an empty with a class of **.progress-bar** and class **progress-bar-*** where * could be **success, info, warning, danger**.
- Add a style attribute with the width expressed as a percentage. Say for example, style="60%"; indicates that the progress bar was at 60%.

Let us see an example below:

```
<div >
   <div
      aria-valuenow="60" aria-valuemin="0" aria-valuemax="100"
      style="width: 90%;">
      <span >90% Complete (Sucess)</span>
   </div>
</div>
<div >
   <div
      aria-valuenow="60" aria-valuemin="0" aria-valuemax="100"
      style="width: 30%;">
      <span >30% Complete (info)</span>
```

```
   </div>
</div>
<div >
   <div
      aria-valuenow="60" aria-valuemin="0" aria-valuemax="100"
      style="width: 20%;">
      <span >20%Complete (warning)</span>
   </div>
</div>
<div >
   <div
      aria-valuenow="60" aria-valuemin="0" aria-valuemax="100"
      style="width: 10%;">
      <span >10% Complete (danger)</span>
   </div>
</div>
```

Striped Progress Bar

To create a striped progress bar:

- Add a with a class of **.progress** and **.progress-striped**.
- Next, inside the above , add an empty with a class of **.progress-bar** and class **progress-bar-*** where * could be **success, info, warning, danger**.
- Add a style attribute with the width expressed as a percentage. Say for example, style="60%"; indicates that the progress bar was at 60%.

Let us see an example below:

```
<div >
   <div
      aria-valuenow="60" aria-valuemin="0" aria-valuemax="100"
      style="width: 90%;">
      <span >90% Complete (Sucess)</span>
   </div>
</div>
<div >
   <div
      aria-valuenow="60" aria-valuemin="0" aria-valuemax="100"
      style="width: 30%;">
      <span >30% Complete (info)</span>
   </div>
</div>
<div >
   <div
      aria-valuenow="60" aria-valuemin="0" aria-valuemax="100"
      style="width: 20%;">
      <span >20%Complete (warning)</span>
   </div>
</div>
<div >
   <div
      aria-valuenow="60" aria-valuemin="0" aria-valuemax="100"
      style="width: 10%;">
      <span >10% Complete (danger)</span>
   </div>
</div>
```

Animated Progress Bar

To create an animated progress bar:

- Add a with a class of **.progress** and **.progress-striped**. Also add class **.active** to **.progress-striped**.
- Next, inside the above , add an empty with a class of **.progress-bar**.
- Add a style attribute with the width expressed as a percentage. Say for example, style="60%"; indicates that the progress bar was at 60%.

This will animate the stripes right to left.

Let us see an example below:

```
<div >
   <div
      aria-valuenow="60" aria-valuemin="0" aria-valuemax="100"
      style="width: 40%;">
      <span >40% Complete</span>
   </div>
</div>
```

Stacked Progress Bar

You can even stack multiple progress bars. Place the multiple progress bars into the same **.progress** to stack them as seen in the following example:

```
<div >
   <div
      aria-valuenow="60" aria-valuemin="0" aria-valuemax="100"
      style="width: 40%;">
      <span >40% Complete</span>
   </div>
   <div
      aria-valuenow="60" aria-valuemin="0" aria-valuemax="100"
      style="width: 30%;">
      <span >30% Complete (info)</span>
   </div>
   <div
      aria-valuenow="60" aria-valuemin="0" aria-valuemax="100"
      style="width: 20%;">
      <span >20%Complete (warning)</span>
   </div>
</div>
```

BOOTSTRAP MEDIA OBJECT

This chapter discusses about Media object. These are abstract object styles for building various types of components *likeblogcomments, Tweets, etc.* that feature a left-aligned or right-aligned image alongside the textual content. The goal of the media object is to make the code for developing these blocks of information drastically shorter.

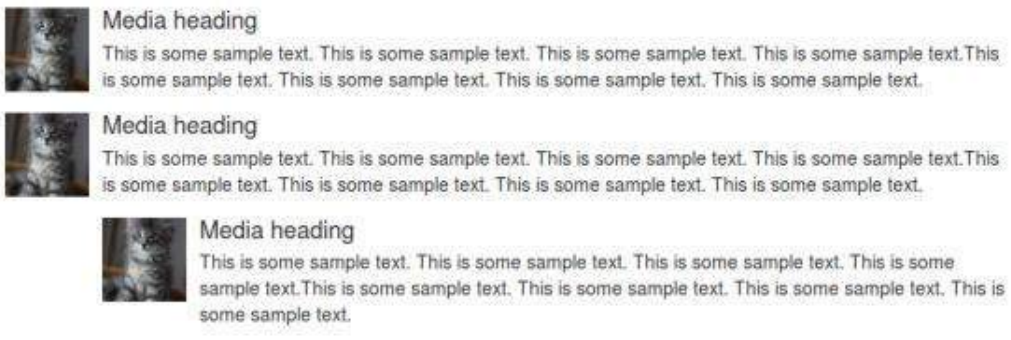
The goal of media objects *lightmarkup, easyextendability* is achieved by applying classes to some of the simple markup. There are two forms to the media object:

- **.media** : This class allows to float a media object *images, video, andaudio* to the left or right of a content block.

- **.media-list** : If you are preparing a list where the items will be part of an unordered list, use a class. useful for comment threads or articles lists.

Let us see an example below of default media object:

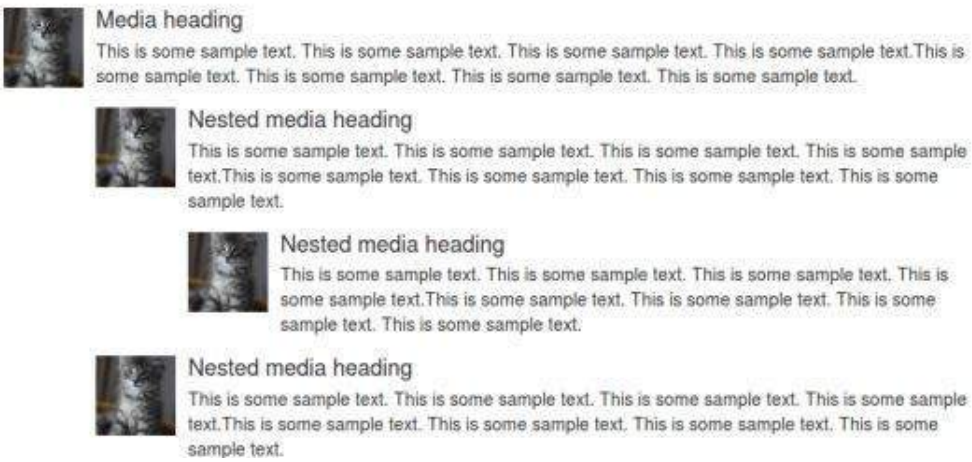
```
<div >
   <a >
      <img
      alt="Media Object">
   </a>
   <div >
      <h4 >Media heading</h4>
      This is some sample text. This is some sample text.
      This is some sample text. This is some sample text.
      This is some sample text. This is some sample text.
      This is some sample text. This is some sample text.
   </div>
</div>
<div >
   <a >
      <img
      alt="Media Object">
   </a>
   <div >
      <h4 >Media heading</h4>
      This is some sample text. This is some sample text.
      This is some sample text. This is some sample text.
      This is some sample text. This is some sample text.
      This is some sample text. This is some sample text.
      <div >
         <a >
            <img
            alt="Media Object">
         </a>
         <div >
            <h4 >Media heading</h4>
            This is some sample text. This is some sample text.
            This is some sample text. This is some sample text.
            This is some sample text. This is some sample text.
            This is some sample text. This is some sample text.
         </div>
      </div>
   </div>
</div>
```



Let us see an example of media list:

```
<ul >
   <li >
      <a >
         <img
         alt="Generic placeholder image">
      </a>
      <div >
         <h4 >Media heading</h4>
         <p>This is some sample text. This is some sample text.
            This is some sample text. This is some sample text.
            This is some sample text. This is some sample text.
            This is some sample text. This is some sample text.</p>
         <!-- Nested media object -->
```

```
         <div >
            <a >
               <img
               alt="Generic placeholder image">
            </a>
            <div >
               <h4 >Nested media heading</h4>
               This is some sample text. This is some sample text.
               This is some sample text. This is some sample text.
               This is some sample text. This is some sample text.
               This is some sample text. This is some sample text.
               <!-- Nested media object -->
               <div >
                  <a >
                     <img
                     alt="Generic placeholder image">
                  </a>
                  <div >
                     <h4 >Nested media heading</h4>
                     This is some sample text. This is some sample text.
                     This is some sample text. This is some sample text.
                     This is some sample text. This is some sample text.
                     This is some sample text. This is some sample text.
                  </div>
               </div>
            </div>
         </div>
         <!-- Nested media object -->
         <div >
            <a >
               <img
               alt="Generic placeholder image">
            </a>
            <div >
               <h4 >Nested media heading</h4>
               This is some sample text. This is some sample text.
               This is some sample text. This is some sample text.
               This is some sample text. This is some sample text.
               This is some sample text. This is some sample text.
            </div>
         </div>
      </div>
   </li>
   <li >
      <a >
         <img
         alt="Generic placeholder image">
      </a>
      <div >
         <h4 >Media heading</h4>
         This is some sample text. This is some sample text.
         This is some sample text. This is some sample text.
         This is some sample text. This is some sample text.
         This is some sample text. This is some sample text.
      </div>
   </li>
</ul>
```



BOOTSTRAP LIST GROUP

The purpose of list group component is to render complex and customized content in lists. To get a basic list group:

- Add the class **.list-group** to element <ul>.
- Add class **.list-group-item** to <li>.

The following example demonstrates this:

```
<ul >
   <li >Free Domain Name Registration</li>
   <li >Free Window Space hosting</li>
   <li >Number of Images</li>
   <li >24*7 support</li>
   <li >Renewal cost per year</li>
</ul>
```

Adding Badges to List Group

We can add the badges component to any list group item and it will automatically be positioned on the right. Just add **** within the <li> element. The following example demonstrates this:

```
<ul >
   <li >Free Domain Name Registration</li>
   <li >Free Window Space hosting</li>
   <li >Number of Images</li>
   <li >
      <span >New</span>
      24*7 support
   </li>
   <li >Renewal cost per year</li>
   <li >
      <span >New</span>
      Disocunt Offer
   </li>
</ul>
```

Linking List Group Items

By using the anchor tags instead of list items, we can link the list groups. We need to use instead of <ul> element. The following example demonstrates this:

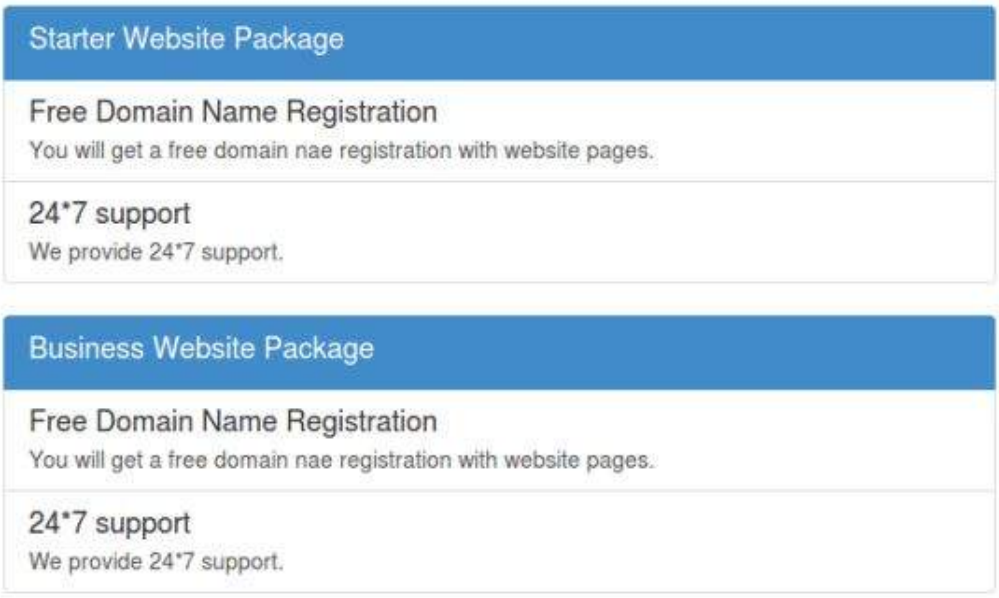
```
<a href="#" >
   Free Domain Name Registration
</a>
<a href="#" >24*7 support</a>
<a href="#" >Free Window Space hosting</a>
<a href="#" >Number of Images</a>
<a href="#" >Renewal cost per year</a>
```

Add Custom Content to List Group

We can add any HTML content to the above linked list groups. The following example demonstrates this:

```
<div >
   <a href="#" >
      <h4 >
         Starter Website Package
      </h4>
   </a>
   <a href="#" >
      <h4 >
         Free Domain Name Registration
      </h4>
      <p >
         You will get a free domain registration with website pages.
      </p>
   </a>
   <a href="#" >
      <h4 >
         24*7 support
      </h4>
      <p >
         We provide 24*7 support.
      </p>
   </a>
</div>
<div >
   <a href="#" >
      <h4 >
         Business Website Package
      </h4>
   </a>
   <a href="#" >
      <h4 >
         Free Domain Name Registration
      </h4>
      <p >
```

```
         You will get a free domain registration with website pages.
      </p>
   </a>
   <a href="#" >
      <h4 >24*7 support</h4>
      <p >We provide 24*7 support.</p>
   </a>
</div>
```



BOOTSTRAP PANELS

This chapter will discuss about Bootstrap panels. Panel components are used when you want to put your DOM component in a box. To get a basic panel, just add class **.panel** to the element. Also add class **.panel-default** to this element as shown in the following example:

```
<div >
   <div >
      This is a Basic panel
   </div>
</div>
```

Panel with Heading

There are two ways to add panel heading:

- Use **.panel-heading** class to easily add a heading container to your panel.
- Use any <h1>-<h6> with a **.panel-title** class to add a pre-styled heading.

The following example demonstrates both the ways:

```
<div >
   <div >
      Panel heading without title
   </div>
   <div >
      Panel content
   </div>
</div>

<div >
   <div >
      <h3 >
```

```
         Panel With title
      </h3>
   </div>
   <div >
      Panel content
   </div>
</div>
```

Panel with Footer

You can add footers to panels, by wrapping buttons or secondary text in a containing class **.panel-footer**. The following example demonstrates this.

```
<div >
   <div >
      This is a Basic panel
   </div>
   <div >Panel footer</div>
</div>
```

> *Panel footers do not inherit colors and borders when using contextual variations as they are not meant to be in the foreground.*

Panel Contextual Alternatives

Use contextual state classes such as, **panel-primary, panel-success, panel-info, panel-warning, panel-danger**, to make a panel more meaningful to a particular context.

```
<div >
   <div >
      <h3 >Panel title</h3>
   </div>
   <div >
      This is a Basic panel
   </div>
</div>
<div >
   <div >
      <h3 >Panel title</h3>
   </div>
   <div >
      This is a Basic panel
   </div>
</div>
<div >
   <div >
      <h3 >Panel title</h3>
   </div>
```

```
   <div >
      This is a Basic panel
   </div>
</div>
<div >
   <div >
      <h3 >Panel title</h3>
   </div>
   <div >
      This is a Basic panel
   </div>
</div>
<div >
   <div >
      <h3 >Panel title</h3>
   </div>
   <div >
      This is a Basic panel
   </div>
</div>
```

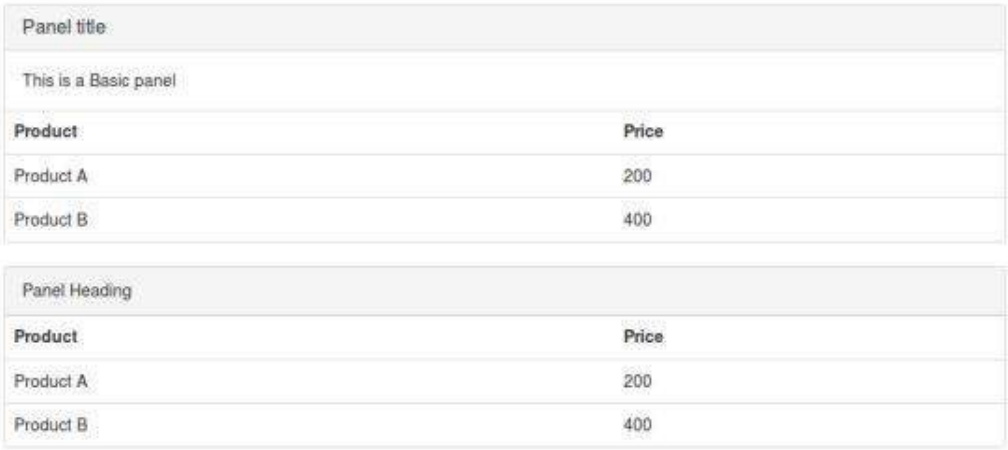
Panel with Tables

To get a non-bordered table within a panel, use the class **.table** within the panel. Suppose there is a containing **.panel-body**, we add an extra border to the top of the table for separation. If there is no containing **.panel-body**, then the component moves from panel header to table without interruption.

The following example demonstrates this:

```
<div >
   <div >
      <h3 >Panel title</h3>
   </div>
   <div >
      This is a Basic panel
   </div>
   <table >
      <th>Product</th><th>Price </th>
      <tr><td>Product A</td><td>200</td></tr>
      <tr><td>Product B</td><td>400</td></tr>
   </table>
</div>
<div >
   <div >Panel Heading</div>
   <table >
      <th>Product</th><th>Price </th>
      <tr><td>Product A</td><td>200</td></tr>
      <tr><td>Product B</td><td>400</td></tr>
```

```
   </table>
</div>
```



Panel with Listgroups

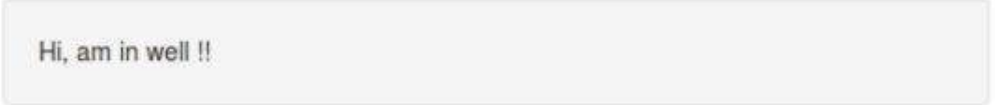
You can include list groups within any panel. Create a panel by adding class **.panel** to the element. Also add class **.panel-default** to this element. Now within this panel include your list groups. You can learn to create a list group from chapter List Groups.

```
<div >
   <div >Panel heading</div>
      <div >
         <p>This is a Basic panel content. This is a Basic panel content.
         This is a Basic panel content.This is a Basic panel content.
         This is a Basic panel content.This is a Basic panel content.
         This is a Basic panel content.
         </p>
   </div>
   <ul >
      <li >Free Domain Name Registration</li>
      <li >Free Window Space hosting</li>
      <li >Number of Images</li>
      <li >24*7 support</li>
      <li >Renewal cost per year</li>
   </ul>
</div>
```

BOOTSTRAP WELLS

A well is a container in that causes the content to appear sunken or an inset effect on the page. To create a well, simply wrap the content that you would like to appear in the well with a containing the class of **.well**. The following example shows a default well:

```
 <div >Hi, am in well !!</div>
```



Sizing

You can change the size of well using the optional classes such as, **well-lg** or **well-lg**. These classes are used in conjunction with **.well** class. These affect the padding, making the well larger or smaller depending on the class.

```
<div >Hi, am in large well !!</div>
<div >Hi, am in small well !!</div>
```

BOOTSTRAP PLUGINS OVERVIEW

The components discussed in the previous chapters under **Layout Components** are just the beginning. Bootstrap comes bundled with 12 jQuery plugins that extend the features and can add more interaction to your site. To get started with the Bootstrap's JavaScript plugins, you don't need to be an advanced JavaScript developer. By utilizing Bootstrap Data API, most of the plugins can be triggered without writing a single line of code.

Bootstrap Plugins can be included on your site in two forms:

- **Individually** : Using Bootstrap's individual **.js* files. Some plugins and CSS components depend on other plugins. If you include plugins individually, make sure to check for these dependencies in the docs.
- Or **compiled** ***allatonce*** : Using *bootstrap.js* or the minified *bootstrap.min.js*.

> *Do not attempt to include both, as both bootstrap.js and bootstrap.min.js contain all plugins in a single file.*

All plugins depend on jQuery. So jQuery must be included before the plugin files. Check bower.json to see which versions of jQuery are supported.

Data Attributes

- All of the Bootstrap plugins are accessible using the included Data API. Hence, you don't need to include a single line of JavaScript to invoke any of the plugin features.
- In some situations it may be desirable to turn this functionality of Data API off. If you need to turn off the Data API, you can unbind the attributes by adding the following line of JavaScript:

```
$(document).off('.data-api')
```

- To turn off a specific/single plugin, just include the plugin's name as a namespace along with the data-api namespace like this:

```
$(document).off('.alert.data-api')
```

Programmatic API

The developers of Bootstrap believe that you should be able to use all of the plugins purely through the JavaScript API. All public APIs are single, chainable methods, and return the collection acted upon say for example:

```
$(".btn.danger").button("toggle").addClass("fat")
```

All methods accept an optional options object, a string which targets a particular method, or

nothing *whichinitiatesapluginwithdefaultbehavior* as shown below:

```
// initialized with defaults
$("#myModal").modal()
 // initialized with no keyboard
$("#myModal").modal({ keyboard: false })
// initializes and invokes show immediately
$("#myModal").modal('show')
```

Each plugin also exposes its raw constructor on a **Constructor** property: *$.fn.popover.Constructor*. If you'd like to get a particular plugin instance, retrieve it directly from an element:

```
 $('[rel=popover]').data('popover').
```

No Conflict

Bootstrap plugins can sometimes be used with other UI frameworks. In these circumstances, namespace collisions can occasionally occur. To overcome this call **.noConflict** on the plugin you wish to revert the value of.

```
// return $.fn.button to previously assigned value
var bootstrapButton = $.fn.button.noConflict()
// give $().bootstrapBtn the Bootstrap functionality
$.fn.bootstrapBtn = bootstrapButton
```

Events

Bootstrap provides custom events for most plugin's unique actions. Generally, these events come in two forms:

- **Infinitive form** : This is triggered at the start of an event. *E.g. show*. Infinitive events provide *preventDefault* functionality. This provides the ability to stop the execution of an action before it starts.

  ```
  $('#myModal').on('show.bs.modal', function (e) {
  // stops modal from being shown
    if (!data) return e.preventDefault()
  })
  ```

- **Past participle form** : This is triggered on the completion of an action. *E.g. shown.*

BOOTSTRAP TRANSITION PLUGIN

The transition plugin provides a simple transition effects.

> *If you want to include this plugin functionality individually, then you will need **transition.js** once alongside the other JS files. Else, as mentioned in the chapter Bootstrap Plugins Overview, you can include bootstrap.js or the minified bootstrap.min.js.*

Transition.js is a basic helper for transitionEnd events as well as a CSS transition emulator. It is used by the other plugins to check for CSS transition support and to catch hanging transitions.

Use Cases

A few examples of the transition plugin are :

- Sliding or fading in modals. You can find an example in the chapter Bootstrap Modal Plugin.
- Fading out tabs. You can find an example in the chapter Bootstrap Tab Plugin.
- Fading out alerts. You can find an example in the chapter Bootstrap Alerts.
- Sliding carousel panes. You can find an example in the chapter Bootstrap Carousel Plugin.

BOOTSTRAP MODAL PLUGIN

A modal is a child window that is layered over its parent window. Typically, the purpose is to display content from a separate source that can have some interaction without leaving the parent window. Child windows can provide information, interaction, or more.

> *If you want to include this plugin functionality individually, then you will need **modal.js**. Else, as mentioned in the chapter Bootstrap Plugins Overview, you can include bootstrap.js or the minified bootstrap.min.js.*

Usage

You can toggle the modal plugin's hidden content:

- **Via data attributes** : Set attribute **data-toggle="modal"** on a controller element, like a button or link, along with a **data-target="#identifier"** or **href="#identifier"** to target a specific modal *withthe* to toggle.
- **Via JavaScript** : Using this technique you can call a modal with with a single line of JavaScript:

```
$('#identifier').modal(options)
```

Example

A static modal window example is shown in the following example:

```
<h2>Example of creating Modals with Twitter Bootstrap</h2>
<!-- Button trigger modal -->
<button
   data-target="#myModal">
   Launch demo modal
</button>

<!-- Modal -->
<div
   aria-labelledby="myModalLabel" aria-hidden="true">
   <div >
      <div >
         <div >
            <button type="button"
               data-dismiss="modal" aria-hidden="true">
                  &times;
            </button>
            <h4 >
               This Modal title
            </h4>
         </div>
         <div >
            Add some text here
         </div>
         <div >
            <button type="button"
               data-dismiss="modal">Close
            </button>
            <button type="button" >
               Submit changes
            </button>
         </div>
      </div><!-- /.modal-content -->
</div><!-- /.modal -->
```

Details of the preceding code:

- To invoke the modal window, you need to have some kind of a trigger. You can use a button or a link. Here we have used a button.
- If you look in the code above, you will see that in the <button> tag, the **data-**

target="#myModal" is the target of the modal that you want to load on the page. This code allows you to create multiple modals on the page and then have different triggers for each of them. Now, to be clear, you don't load multiple modals at the same time, but you can create many on the pages to be loaded at different times.

- There are two classes to take note of in the modal:
    - The first is **.modal**, which is simply identifying the content of the as a modal.
    - And second is the **.fade** class. When the modal is toggled, it will cause the content to fade in and out.
- **aria-labelledby="myModalLabel"**, attribute reference the modal title.
- The attribute **aria-hidden="true"** is used to keep the Modal Window invisible till a trigger comes *likeaclickontheassociatedbutton*.
- , modal-header is the class to define style for the header of the modal window.
- , is a CSS class close that sets style for the Close button of the modal window.
- **data-dismiss="modal"**, is a custom HTML5 data attribute. Here it is used to close the modal window.
- , is a CSS class of Bootstrap CSS to set style for body of the modal window.
- , is a CSS class of Bootstrap CSS to set style for footer of the modal window.
- **data-toggle="modal"**, HTML5 custom data attribute data-toggle is used to open the modal window.

Options

There are certain options which can be passed via data attributes or JavaScript to customize the look and feel of the Modal Window. Following table lists the options:

Option Name	Type/Default Value	Data attribute name	Description
backdrop	boolean or the string 'static' *Default: true*	data-backdrop	Specify static for a backdrop, if you don't want the modal to be closed when the user clicks outside of the modal.
keyboard	boolean *Default: true*	data-keyboard	Closes the modal when escape key is pressed; set to false to disable.
show	boolean *Default: true*	data-show	Shows the modal when initialized.
remote	path *Default: false*	data-remote	Using the jQuery *.load* method, inject content into the modal body. If an href with a valid URL is added, it will load that content. An example of this is shown below:

```
<a data-toggle="modal" href="remote.html" data-
target="#modal">Click me</a>
```

Methods

Here are some useful methods that can be used with modal.

Method	Description	Example
Options: .modal*options*	Activates your content as a modal. Accepts an optional options object.	`$('#identifier').modal({ keyboard: false })`
Toggle : .modal*'toggle'*	Manually toggles a modal.	`$('#identifier').modal('toggle')`
Show: .modal*'show'*	Manually opens a modal.	`$('#identifier').modal('show')`
Hide : .modal *'hide'*	Manually hides a modal.	`$('#identifier').modal('hide')`

Example

The following example demonstrates the usage of methods:

```
<h2>Example of using methods of Modal Plugin</h2>

<!-- Button trigger modal -->
<button >
   Launch demo modal
</button>

<!-- Modal -->
<div
   aria-labelledby="myModalLabel" aria-hidden="true">
   <div >
      <div >
         <div >
            <button type="button"
               aria-hidden="true">×
            </button>
            <h4 >
               This Modal title
            </h4>
         </div>
         <div >
            Press ESC button to exit.
         </div>
         <div >
            <button type="button"
               data-dismiss="modal">Close
            </button>
            <button type="button" >
               Submit changes
            </button>
         </div>
      </div><!-- /.modal-content -->
   </div><!-- /.modal-dialog -->
</div><!-- /.modal -->
<script>
   $(function () { $('#myModal').modal({
      keyboard: true
   })});
</script>
```

Just click the Esc button and the modal window exits.

Events

Following table lists the events to work with modal. These events may be used to hook into the function.

Event	Description	Example
show.bs.modal	Fired after the show method is called.	`$('#identifier').on('show.bs.modal', function () { // do something… })`
shown.bs.modal	Fired when the modal has been made visible to the user *willwaitforCSStransitionstocomplete*.	`$('#identifier').on('shown.bs.modal', function () { // do something… })`
hide.bs.modal	Fired when the hide instance method has been called.	`$('#identifier').on('hide.bs.modal', function () { // do something… })`
hidden.bs.modal	Fired when the modal has finished being hidden from the user.	`$('#identifier').on('hidden.bs.modal', function () { // do something… })`

Example

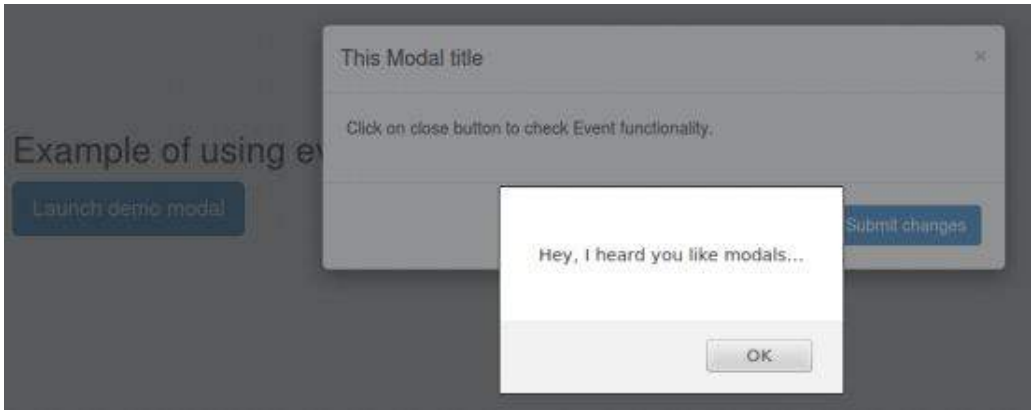
The following example demonstrates the usage of events:

```
<h2>Example of using events of Modal Plugin</h2>

<!-- Button trigger modal -->
<button >
   Launch demo modal
</button>

<!-- Modal -->
<div
   aria-labelledby="myModalLabel" aria-hidden="true">
   <div >
      <div >
         <div >
            <button type="button"
               aria-hidden="true">×
            </button>
            <h4 >
               This Modal title
            </h4>
         </div>
```

```
         <div >
            Click on close button to check Event functionality.
         </div>
         <div >
            <button type="button"
               data-dismiss="modal">
               Close
            </button>
            <button type="button" >
               Submit changes
            </button>
         </div>
      </div><!-- /.modal-content -->
   </div><!-- /.modal-dialog -->
</div><!-- /.modal -->
<script>
   $(function () { $('#myModal').modal('hide')})});
</script>
<script>
   $(function () { $('#myModal').on('hide.bs.modal', function () {
      alert('Hey, I heard you like modals...');})
   });
</script>
```



As seen in the above screen, if you click on the *Close* button i.e *hide* event, an alert message is displayed.

BOOTSTRAP DROPDOWN PLUGIN

Using Dropdown plugin you can add dropdown menus to any components like navbars, tabs, pills and buttons.

> *If you want to include this plugin functionality individually, then you will need **dropdown.js**. Else, as mentioned in the chapter Bootstrap Plugins Overview, you can include bootstrap.js or the minified bootstrap.min.js.*

Usage

You can toggle the dropdown plugin's hidden content:

- **Via data attributes** : Add **data-toggle="dropdown"** to a link or button to toggle a dropdown as shown below:

```
<div >
  <a data-toggle="dropdown" href="#">Dropdown trigger</a>
  <ul >
    ...
  </ul>
</div>
```

If you need to keep links intact *whichisusefulifthebrowserisnotenablingJavaScript*, use the **data-target** attribute instead of **href="#"**:

```
<div >
  <a >
    Dropdown <span ></span>
  </a>


  <ul >
    ...
  </ul>
</div>
```

- **Via JavaScript** : To call the dropdown toggle via JavaScript, use the following method:

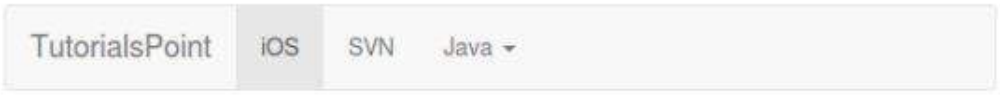
```
$('.dropdown-toggle').dropdown()
```

Example

Within Navbar

The following example demonstrates the usage of dropdown menu within a navbar:

```
<nav >
   <div >
      <a >TutorialsPoint</a>
   </div>
   <div>
      <ul >
         <li >iOS</a></li>
         <li><a href="#">SVN</a></li>
         <li >
            <a href="#" >
               Java
               <b ></b>
            </a>
            <ul >
               <li><a href="#">jmeter</a></li>
               <li><a href="#">EJB</a></li>
               <li><a href="#">Jasper Report</a></li>
               <li ></li>
               <li><a href="#">Separated link</a></li>
               <li ></li>
               <li><a href="#">One more separated link</a></li>
            </ul>
         </li>
      </ul>
   </div>
</nav>
```



Within Tabs

The following example demonstrates the usage of dropdown menu within tabs:

```
<p>Tabs With Dropdown Example</p>
<ul >
   <li >Home</a></li>
   <li><a href="#">SVN</a></li>
   <li><a href="#">iOS</a></li>
   <li><a href="#">VB.Net</a></li>
   <li >
      <a >
         Java <span ></span>
      </a>
      <ul >
```

```
         <li><a href="#">Swing</a></li>
         <li><a href="#">jMeter</a></li>
         <li><a href="#">EJB</a></li>
         <li ></li>
         <li><a href="#">Separated link</a></li>
      </ul>
   </li>
   <li><a href="#">PHP</a></li>
</ul>
```

Options

There are no options.

Methods

The dropdown toggle has a simple method to show or hide the dropdown.

```
$().dropdown('toggle')
```

Example

The following example demonstrates the usage of dropdown plugin method.

```
<nav >
   <div >
      <a >TutorialsPoint</a>
   </div>

   <div >
      <ul >
         <li >iOS</a></li>
         <li><a href="#">SVN</a></li>
         <li >
            <a href="#" >Java <b
               ></b></a>
               <ul >
                  <li><a >
                     jmeter</a>
                  </li>
                  <li><a href="#">EJB</a></li>
                  <li><a href="#">Jasper Report</a></li>
                  <li ></li>
                  <li><a href="#">Separated link</a></li>
                  <li ></li>
                  <li><a href="#">One more separated link</a></li>
               </ul>
            </li>
         </ul>
      </div>
   </nav>
</div>
<script>
   $(function(){
      $(".dropdown-toggle").dropdown('toggle');
      });
</script>
```

BOOTSTRAP SCROLLSPY PLUGIN

The Scrollspy *autoupdatingnav* plugin allows you to target sections of the page based on the scroll position. In its basic implementation, as you scroll, you can add **.active** classes to the navbar based on the scroll position.

> *If you want to include this plugin functionality individually, then you will need **scrollspy.js**. Else, as mentioned in the chapter Bootstrap Plugins Overview, you can include bootstrap.js or the minified bootstrap.min.js.*

Usage

You can add scrollspy behavior to your topbar navigation:

- **Via data attributes** : add **data-spy="scroll"** to the element you want to spy on *typicallythebody*. Then add attribute **data-target** with the ID or class of the parent element of any Bootstrap **.nav** component. For this to work, you must have elements in the body of the page that have matching IDs of the links that you are spying on.

```
<body data-spy="scroll" data-target=".navbar-example">
  ...
  <div >
    <ul >
      ...
    </ul>
  </div>
  ...
</body>
```

- **Via JavaScript** : You can invoke the scrollspy with JavaScript instead of using the data attributes, by selecting the element to spy on, and then invoking the **.scrollspy** function:

```
$('body').scrollspy({ target: '.navbar-example' })
```

Example

The following example shows the use of scrollspy plugin via data attributes:

```
<nav  >
   <div >
      <button
         data-target=".bs-js-navbar-scrollspy">
         <span >Toggle navigation</span>
         <span ></span>
         <span ></span>
         <span ></span>
      </button>
      <a >Tutorial Name</a>
   </div>
   <div >
      <ul >
         <li><a href="#ios">iOS</a></li>
         <li><a href="#svn">SVN</a></li>
```

```
         <li >
            <a href="#"
               data-toggle="dropdown">Java
               <b ></b>
            </a>
            <ul
               aria-labelledby="navbarDrop1">
               <li><a href="#jmeter" tabindex="-1">jmeter</a></li>
               <li><a href="#ejb" tabindex="-1">ejb</a></li>
               <li ></li>
               <li><a href="#spring" tabindex="-1">spring</a></li>
            </ul>
         </li>
      </ul>
   </div>
</nav>
<div data-spy="scroll" data-target="#navbar-example" data-offset="0"
   style="height:200px;overflow:auto; position: relative;">
   <h4 >iOS</h4>
   <p>iOS is a mobile operating system developed and distributed by Apple
      Inc. Originally released in 2007 for the iPhone, iPod Touch, and Apple
      TV. iOS is derived from OS X, with which it shares the Darwin
      foundation. iOS is Apple's mobile version of the OS X operating system
      used on Apple computers.
   </p>
   <h4 >SVN</h4>
   <p>Apache Subversion which is often abbreviated as SVN, is a software
      versioning and revision control system distributed under an open source
      license. Subversion was created by CollabNet Inc. in 2000, but now it
      is developed as a project of the Apache Software Foundation, and as
      such is part of a rich community of developers and users.
   </p>
   <h4 >jMeter</h4>
   <p>jMeter is an Open Source testing software. It is 100% pure Java
      application for load and performance testing.
   </p>
   <h4 >EJB</h4>
   <p>Enterprise Java Beans (EJB) is a development architecture for building
      highly scalable and robust enterprise level applications to be deployed
      on J2EE compliant Application Server such as JBOSS, Web Logic etc.
   </p>
   <h4 >Spring</h4>
   <p>Spring framework is an open source Java platform that provides
      comprehensive infrastructure support for developing robust Java
      applications very easily and very rapidly.
   </p>
   <p>Spring framework was initially written by Rod Johnson and was first
      released under the Apache 2.0 license in June 2003.
   </p>
</div>
```

Options

Options can be passed via data attributes or JavaScript. Following table lists the options:

Option Name	Type/Default Value	Data attribute name	Description
offset	number *Default: 10*	data-offset	Pixels to offset from top when calculating position of scroll.

Methods

.scrollspy'*refresh*' : When calling the scrollspy via the JavaScript method, you need to call the **.refresh** method to update the DOM. This is helpful if any elements of the DOM have changed i.e if you have added or removed some elements. Following would be the syntax to use this method.

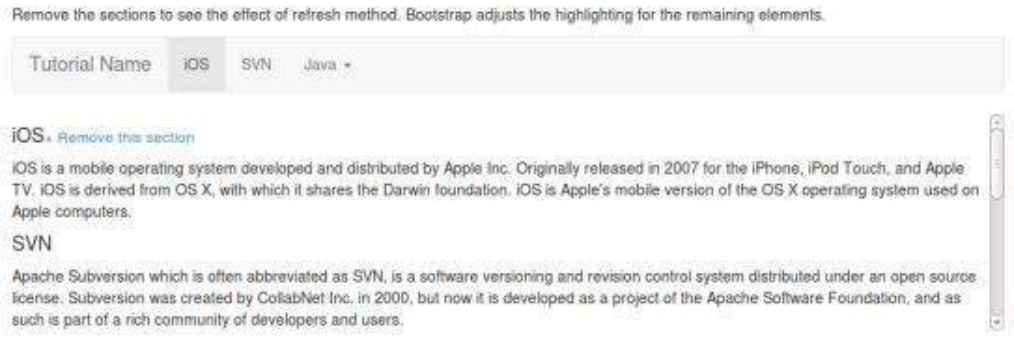
```
$('[data-spy="scroll"]').each(function () {
  var $spy = $(this).scrollspy('refresh')
})
```

Example

The following example demonstrates the use of**.scrollspy**'*refresh*' method:

```
<nav  >
   <div >
      <button
         data-target=".bs-js-navbar-scrollspy">
         <span >Toggle navigation</span>
         <span ></span>
         <span ></span>
         <span ></span>
      </button>
      <a >Tutorial Name</a>
   </div>
   <div >
      <ul >
         <li >iOS</a></li>
         <li><a href="#svn">SVN</a></li>
         <li >
            <a href="#"
               data-toggle="dropdown">Java
               <b ></b>
            </a>
            <ul
               aria-labelledby="navbarDrop1">
               <li><a href="#jmeter" tabindex="-1">jmeter</a></li>
               <li><a href="#ejb" tabindex="-1">ejb</a></li>
               <li ></li>
               <li><a href="#spring" tabindex="-1">spring</a></li>
            </ul>
         </li>
      </ul>
   </div>
</nav>
<div data-spy="scroll" data-target="#myScrollspy" data-offset="0"
   style="height:200px;overflow:auto; position: relative;">
   <div >
      <h4 >
         &times; Remove this section</a></small>
      </h4>
      <p>iOS is a mobile operating system developed and distributed by
      Apple Inc. Originally released in 2007 for the iPhone, iPod Touch, and
      Apple TV. iOS is derived from OS X, with which it shares the Darwin
      foundation. iOS is Apple's mobile version of the OS X operating system
      used on Apple computers.</p>
   </div>
   <div >
      <h4 >SVN<small></small></h4>
      <p>Apache Subversion which is often abbreviated as SVN, is a software
      versioning and revision control system distributed under an open source
      license. Subversion was created by CollabNet Inc. in 2000, but
      now it is developed as a project of the Apache Software Foundation,
      and as such is part of a rich community of developers and users.</p>
   </div>
   <div >
      <h4 >
      &times; Remove this section</a></small>
      </h4>
```

```
      <p>jMeter is an Open Source testing software. It is 100% pure Java
      application for load and performance testing.</p>
   </div>
   <div >
      <h4 >EJB</h4>
      <p>Enterprise Java Beans (EJB) is a development architecture for
         building highly scalable and robust enterprise level applications
         to be deployed on J2EE compliant Application Server such as
         JBOSS, Web Logic etc.</p>
   </div>
   <div >
      <h4 >Spring</h4>
      <p>Spring framework is an open source Java platform that provides
         comprehensive infrastructure support for developing robust Java
         applications very easily and very rapidly.</p>
      <p>Spring framework was initially written by Rod Johnson and was first
         released under the Apache 2.0 license in June 2003.</p>
   </div>
</div>
<script type="text/javascript">
   $(function(){
   removeSection = function(e) {
      $(e).parents(".section").remove();
      $('[data-spy="scroll"]').each(function () {
         var $spy = $(this).scrollspy('refresh')
      });
   }
   $("#myScrollspy").scrollspy();
});
</script>
```



Events

The following table lists the events to work with scrollspy. This event may be used to hook into the function.

Event	Description	Example

activate.bs.scrollspy	This event fires whenever a new item becomes activated by the scrollspy.	`$('#myScrollspy').on('activate.bs.scrollspy', function () {`
		`  // do something…`
		`})`

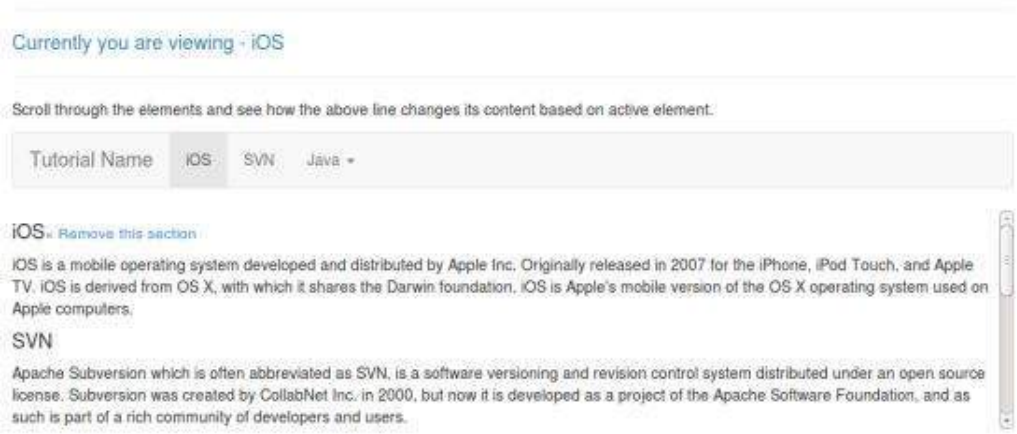
Example

The following example demonstrates the use of**activate.bs.scrollspy** event:

```
<nav  >
   <div >
      <button
         data-target=".bs-js-navbar-scrollspy">
         <span >Toggle navigation</span>
         <span ></span>
         <span ></span>
         <span ></span>
```

```
      </button>
      <a >Tutorial Name</a>
   </div>
   <div >
      <ul >
         <li >iOS</a></li>
         <li><a href="#svn">SVN</a></li>
         <li >
            <a href="#"
               data-toggle="dropdown">
               Java <b ></b>
            </a>
            <ul
               aria-labelledby="navbarDrop1">
               <li><a href="#jmeter" tabindex="-1">jmeter</a></li>
               <li><a href="#ejb" tabindex="-1">ejb</a></li>
               <li ></li>
               <li><a href="#spring" tabindex="-1">spring</a></li>
            </ul>
         </li>
      </ul>
   </div>
</nav>
<div data-spy="scroll" data-target="#myScrollspy" data-offset="0"
   style="height:200px;overflow:auto; position: relative;">
   <div >
      <h4 >
         &times; Remove this section</a></small>
      </h4>
      <p>iOS is a mobile operating system developed and distributed by
         Apple Inc. Originally released in 2007 for the iPhone, iPod Touch,
         and Apple TV. iOS is derived from OS X, with which it shares
         the Darwin foundation. iOS is Apple's mobile version of the OS X
         operating system used on Apple computers.</p>
   </div>
   <div >
      <h4 >SVN<small></small></h4>
      <p>Apache Subversion which is often abbreviated as SVN, is a software
         versioning and revision control system distributed under an open
         source license. Subversion was created by CollabNet Inc. in 2000,
         but now it is developed as a project of the Apache Software
         Foundation, and as such is part of a rich community of developers
         and users.</p>
   </div>
   <div >
      <h4 >
         &times; Remove this section</a></small>
      </h4>
      <p>jMeter is an Open Source testing software. It is 100% pure Java
         application for load and performance testing.</p>
   </div>
   <div >
      <h4 >EJB</h4>
      <p>Enterprise Java Beans (EJB) is a development architecture for
         building highly scalable and robust enterprise level applications
         to be deployed on J2EE compliant Application Server such as JBOSS,
         Web Logic etc.</p>
   </div>
   <div >
      <h4 >Spring</h4>
      <p>Spring framework is an open source Java platform that provides
         comprehensive infrastructure support for developing robust Java
         applications very easily and very rapidly.</p>
      <p>Spring framework was initially written by Rod Johnson and was
         first released under the Apache 2.0 license in June 2003.</p>
   </div>
</div>
<script type="text/javascript">
   $(function(){
      removeSection = function(e) {
      $(e).parents(".section").remove();
      $('[data-spy="scroll"]').each(function () {
          var $spy = $(this).scrollspy('refresh')
      });
```

```
   }
   $("#myScrollspy").scrollspy();
   $('#myScrollspy').on('activate.bs.scrollspy', function () {
      var currentItem = $(".nav li.active > a").text();
      $("#activeitem").html("Currently you are viewing - " + currentItem);
   })
});
</script>
```



BOOTSTRAP TAB PLUGIN

Tabs were introduced in the chapter Bootstrap Navigation Elements. By combining a few data attributes, you can easily create a tabbed interface. With this plugin you can transition through panes of local content in tabs or pills, even via dropdown menus.

> *If you want to include this plugin functionality individually, then you will need **tab.js**. Else, as mentioned in the chapter Bootstrap Plugins Overview, you can include bootstrap.js or the minified bootstrap.min.js.*

Usage

You can enable tabs in the following two ways:

- **Via data attributes** : you need to add **data-toggle="tab"** or **data-toggle="pill"** to the anchors.

  Adding the **nav** and **nav-tabs** classes to the tab **ul** will apply the Bootstrap tab styling, while adding the**nav** and **nav-pills** classes will apply pill styling.

```
<ul >
   <li><a href="#identifier" data-toggle="tab">Home</a></li>
...
</ul>
```

- **Via JavaScript** : you can enable tabs using Javscript as below :

```
$('#myTab a').click(function (e) {
  e.preventDefault()
  $(this).tab('show')
})
```

  Here's an example of different ways to activate individual tabs:

```
// Select tab by name
$('#myTab a[href="#profile"]').tab('show')

// Select first tab
$('#myTab a:first').tab('show')

// Select last tab
$('#myTab a:last').tab('show')
```

```
// Select third tab (0-indexed)
$('#myTab li:eq(2) a').tab('show')
```

Fade Effect

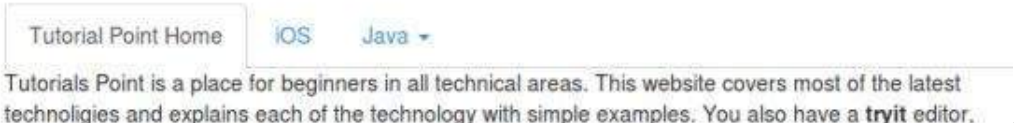
To get a fade effect for tabs, add **.fade** to each **.tab-pane**. The first tab pane must also have **.in** to properly fade in initial content:

```
<div >
  <div >...</div>
  <div >...</div>
  <div >...</div>
  <div >...</div>
</div>
```

Example

An example of tab plugin using data attributes and fade effect is as shown in the following example:

```
<ul  >
   <li >
      <a href="#home" data-toggle="tab">
         Tutorial Point Home
      </a>
   </li>
   <li><a href="#ios" data-toggle="tab">iOS</a></li>
   <li >
      <a href="#"
         data-toggle="dropdown">Java
         <b ></b>
      </a>
      <ul >
         <li><a href="#jmeter" tabindex="-1" data-toggle="tab">jmeter</a></li>
         <li><a href="#ejb" tabindex="-1" data-toggle="tab">ejb</a></li>
      </ul>
   </li>
</ul>
<div  >
   <div >
      <p>Tutorials Point is a place for beginners in all technical areas.
         This website covers most of the latest technoligies and explains
         each of the technology with simple examples. You also have a
         <b>tryit</b> editor, wherein you can edit your code and
         try out different possibilities of the examples.</p>
   </div>
   <div >
      <p>iOS is a mobile operating system developed and distributed by Apple
         Inc. Originally released in 2007 for the iPhone, iPod Touch, and
         Apple TV. iOS is derived from OS X, with which it shares the
         Darwin foundation. iOS is Apple's mobile version of the
         OS X operating system used on Apple computers.</p>
   </div>
   <div >
      <p>jMeter is an Open Source testing software. It is 100% pure
      Java application for load and performance testing.</p>
   </div>
   <div >
      <p>Enterprise Java Beans (EJB) is a development architecture
         for building highly scalable and robust enterprise level
         applications to be deployed on J2EE compliant
         Application Server such as JBOSS, Web Logic etc.
      </p>
   </div>
</div>
```



wherein you can edit your code and try out different possibilities of the examples.

Methods

.$().tab : This method activates a tab element and content container. Tab should have either a **data-target** or an **href** targeting a container node in the DOM.

```
<ul >
  <li >Home</a></li>
  .....
</ul>
<div >
  <div >...</div>
  .....
</div>
<script>
  $(function () {
    $('#myTab a:last').tab('show')
  })
</script>
```

Example

The following example shows the use of tab plugin method **.tab**. Here in the example the second tab *iOS* is activated:

```
<ul  >
   <li >
      Tutorial Point Home</a>
   </li>
   <li><a href="#ios" data-toggle="tab">iOS</a></li>
   <li >
      <a href="#"
         data-toggle="dropdown">Java <b ></b>
      </a>
      <ul >
         <li><a href="#jmeter" tabindex="-1" data-toggle="tab">
            jmeter</a>
         </li>
         <li><a href="#ejb" tabindex="-1" data-toggle="tab">
            ejb</a>
         </li>
      </ul>
   </li>
</ul>
<div  >
   <div >
      <p>Tutorials Point is a place for beginners in all technical areas.
         This website covers most of the latest technoligies and explains
         each of the technology with simple examples. You also have a
         <b>tryit</b> editor, wherein you can edit your code and
         try out different possibilities of the examples.</p>
   </div>
   <div >
      <p>iOS is a mobile operating system developed and distributed by Apple
         Inc. Originally released in 2007 for the iPhone, iPod Touch, and
         Apple TV. iOS is derived from OS X, with which it shares the
         Darwin foundation. iOS is Apple's mobile version of the
         OS X operating system used on Apple computers.</p>
   </div>
   <div >
      <p>jMeter is an Open Source testing software. It is 100% pure
      Java application for load and performance testing.</p>
   </div>
   <div >
      <p>Enterprise Java Beans (EJB) is a development architecture
         for building highly scalable and robust enterprise level
         applications to be deployed on J2EE compliant
         Application Server such as JBOSS, Web Logic etc.
      </p>
   </div>
```

```
</div>
<script>
   $(function () {
      $('#myTab li:eq(1) a').tab('show');
   });
</script>
```

Events

Following table lists the events to work with tab plugin. This event may be used to hook into the function.

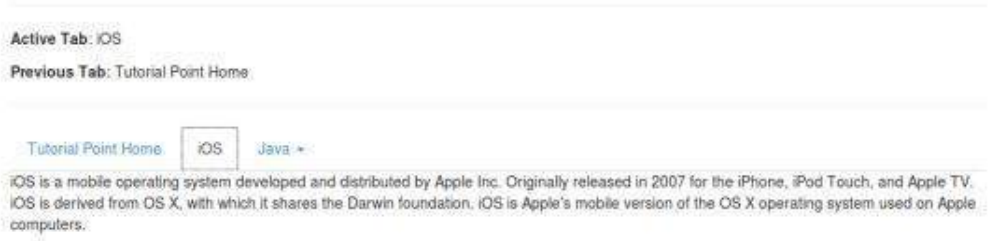
Event	Description	Example
show.bs.tab	This event fires on tab show, but before the new tab has been shown. Use **event.target** and **event.relatedTarget** to target the active tab and the previous active tab *ifavailable* respectively.	`$('a[data-toggle="tab"]').on('show.bs.tab', function (e) { e.target // activated tab e.relatedTarget // previous tab })`
shown.bs.tab	This event fires on tab show after a tab has been shown. Use **event.target** and **event.relatedTarget** to target the active tab and the previous active tab *ifavailable* respectively.	`$('a[data-toggle="tab"]').on('shown.bs.tab', function (e) { e.target // activated tab e.relatedTarget // previous tab })`

Example

The following example shows use of tab plugin events. Here in the example we will display the current and previous visited tabs:

```
<hr>
   <p ><strong>Active Tab</strong>: <span></span></p>
   <p ><strong>Previous Tab</strong>: <span></span></p>
<hr>
<ul  >
   <li >
      Tutorial Point Home</a></li>
   <li><a href="#ios" data-toggle="tab">iOS</a></li>
   <li >
      <a href="#"
         data-toggle="dropdown">
         Java <b ></b></a>
      <ul >
         <li><a href="#jmeter" tabindex="-1" data-toggle="tab">jmeter</a></li>
         <li><a href="#ejb" tabindex="-1" data-toggle="tab">ejb</a></li>
      </ul>
   </li>
</ul>
<div  >
   <div >
      <p>Tutorials Point is a place for beginners in all technical areas.
         This website covers most of the latest technoligies and explains
         each of the technology with simple examples. You also have a
         <b>tryit</b> editor, wherein you can edit your code and
         try out different possibilities of the examples.</p>
   </div>
```

```
   <div >
      <p>iOS is a mobile operating system developed and distributed by Apple
         Inc. Originally released in 2007 for the iPhone, iPod Touch, and
         Apple TV. iOS is derived from OS X, with which it shares the
         Darwin foundation. iOS is Apple's mobile version of the
         OS X operating system used on Apple computers.</p>
   </div>
   <div >
      <p>jMeter is an Open Source testing software. It is 100% pure
      Java application for load and performance testing.</p>
   </div>
   <div >
      <p>Enterprise Java Beans (EJB) is a development architecture
         for building highly scalable and robust enterprise level
         applications to be deployed on J2EE compliant
         Application Server such as JBOSS, Web Logic etc.
      </p>
   </div>
</div>
<script>
   $(function(){
      $('a[data-toggle="tab"]').on('shown.bs.tab', function (e) {
      // Get the name of active tab
      var activeTab = $(e.target).text();
      // Get the name of previous tab
      var previousTab = $(e.relatedTarget).text();
      $(".active-tab span").html(activeTab);
      $(".previous-tab span").html(previousTab);
   });
});
</script>
```



BOOTSTRAP TOOLTIP PLUGIN

Tooltips are useful when you need to describe a link. The plugin was inspired by *jQuery.tipsy* plugin written by *Jason Frame*. Tooltips have since been updated to work without images, animate with a CSS animation, and data-attributes for local title storage.

> *If you want to include this plugin functionality individually, then you will need* ***tooltip.js****. Else, as mentioned in the chapter Bootstrap Plugins Overview, you can include bootstrap.js or the minified bootstrap.min.js.*

Usage

The tooltip plugin generates content and markup on demand, and by default places tooltips after their trigger element. You can add tooltips in the following two ways:

- **Via data attributes** : To add a tooltip, add **data-toggle="tooltip"** to an anchor tag. The title of the anchor will be the text of a tooltip. By default, tooltip is set to top by the plugin.

```
<a href="#" data-toggle="tooltip" title="Example tooltip">Hover over me</a>
```

- **Via JavaScript** : Trigger the tooltip via JavaScript:

```
$('#identifier').tooltip(options)
```

> *Tooltip plugin is NOT only-css plugins like dropdown or other plugins discussed in previous chapters. To use this plugin you MUST activate it using jquery readjavascript. To enable all the tooltips on your page just use this script:*
>
> ```
> $(function () { $("[data-toggle='tooltip']").tooltip(); });
> ```

Example

The following example demonstrates the use of tooltip plugin via data attributes.

```
<h4>Tooltip examples for anchors</h4>
This is a <a href="#"
   title="Tooltip on left">
   Default Tooltip
</a>.
This is a <a href="#"
   data-placement="left" title="Tooltip on left">
   Tooltip on Left
</a>.
This is a <a href="#" data-toggle="tooltip" data-placement="top"
   title="Tooltip on top">
   Tooltip on Top
</a>.
This is a <a href="#" data-toggle="tooltip" data-placement="bottom"
   title="Tooltip on bottom">
   Tooltip on Bottom
</a>.
This is a <a href="#" data-toggle="tooltip" data-placement="right"
   title="Tooltip on right">
   Tooltip on Right
</a>

<br>

<h4>Tooltip examples for buttons</h4>
<button type="button"
   title="Tooltip on left">
   Default Tooltip
</button>
<button type="button"
   data-placement="left" title="Tooltip on left">
   Tooltip on left
</button>
<button type="button"
   data-placement="top" title="Tooltip on top">
   Tooltip on top
</button>
<button type="button"
   data-placement="bottom" title="Tooltip on bottom">
   Tooltip on bottom
</button>
<button type="button"
   data-placement="right" title="Tooltip on right">
   Tooltip on right
</button>
<script>
   $(function () { $("[data-toggle='tooltip']").tooltip(); });
</script>
```

Options

There are certain options which can be added via the Bootstrap Data API or invoked via JavaScript. Following table lists the options:

Option Name	Type/Default Value	Data attribute name	Description
animation	boolean *Default: true*	data-animation	Applies a CSS fade transition to the tooltip.
html	boolean *Default: false*	data-html	Inserts HTML into the tooltip. If false, jQuery's text method will be used to insert content into the dom. Use text if you're worried about XSS attacks.
placement	string|function *Default: top*	data-placement	Specifies how to position the tooltip *i. e.* , *top|bottom|left|right|auto*. When *auto* is specified, it will dynamically reorient the tooltip. For example, if placement is "auto left", the tooltip will display to the left when possible, otherwise it will display right.
selector	string *Default: false*	data-selector	If a selector is provided, tooltip objects will be delegated to the specified targets.
title	string | function *Default:* ''	data-title	The title option is the default title value if the *title* attribute isn't present.
trigger	string *Default:* '*hover focus*'	data-trigger	Defines how the tooltip is triggered: **click| hover | focus | manual**. You may pass multiple triggers; separate them with a space.
content	string | function *Default:* ''	data-content	Default content value if *data-content* attribute isn't present

delay	number | object
	Default: 0
		data-delay	Delays showing and hiding the tooltip in 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 *Default: false*	data-container	Appends the tooltip to a specific element. Example: container: 'body'

Methods

The following are some useful methods for tooltips:

Method	Description	Example
Options : .tooltip *options*	Attaches a tooltip handler to an element collection.	`$().tooltip(options)`

Toggle : .tooltip *'toggle'*	Toggles an element's tooltip.	`$('#element').tooltip('toggle')`
Show : .tooltip *'show'*	Reveals an element's tooltip.	`$('#element').tooltip('show')`
Hide : .tooltip *'hide'*	Hides an element's tooltip.	`$('#element').tooltip('hide')`
Destroy : .tooltip *'destroy'*	Hides and destroys an element's tooltip.	`$('#element').tooltip('destroy')`

Example

The following example demonstrates the use of tooltip plugin via data attributes.

```
<div style="padding: 100px 100px 10px;">
   This is a <a href="#"
      title="show">Tooltip on method show
   </a>.

   This is a <a href="#"
      data-placement="left" title="hide">Tooltip on method hide
   </a>.

   This is a <a href="#"
      data-placement="top" title="destroy">Tooltip on method destroy
   </a>.

   This is a <a href="#"
      data-placement="bottom" title="toggle">Tooltip on method toggle
   </a>.
   <br><br><br><br><br><br>
   <p  >
      This is a <a href="#" data-toggle="tooltip" title="<h2>'am Header2
         </h2>">Tooltip on method options
      </a>.
   </p>

   <script>
      $(function () { $('.tooltip-show').tooltip('show');});
      $(function () { $('.tooltip-hide').tooltip('hide');});
      $(function () { $('.tooltip-destroy').tooltip('destroy');});
      $(function () { $('.tooltip-toggle').tooltip('toggle');});
      $(function () { $(".tooltip-options a").tooltip({html : true });
      });
   </script>
<div>
```

Events

Following table lists the events to work with tooltip plugin. This event may be used to hook into the function.

Event	Description	Example
show.bs.tooltip	This event fires immediately	`$('#myTooltip').on('show.bs.tooltip',`

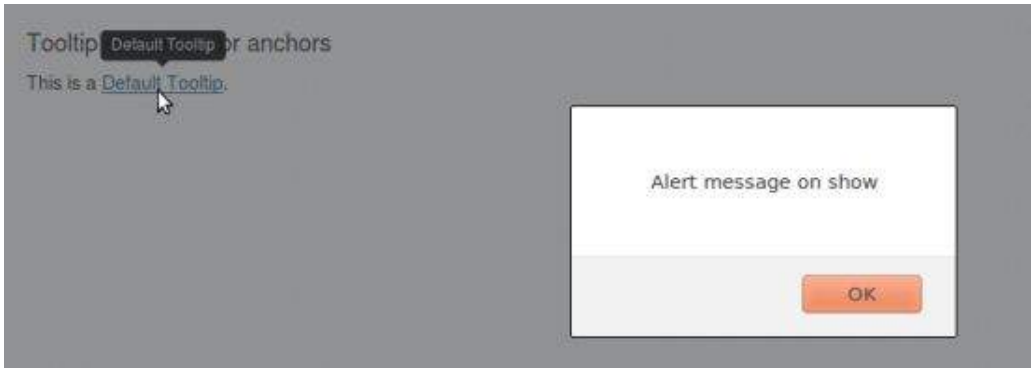
	when the show instance method is called.	`function () { // do something… })`
shown.bs.tooltip	This event is fired when the tooltip has been made visible to the user *willwaitforCSStransitionstocomplete*..	`$('#myTooltip').on('shown.bs.tooltip', function () { // do something… })`
hide.bs.tooltip	This event is fired immediately when the hide instance method has been called.	`$('#myTooltip').on('hide.bs.tooltip', function () { // do something… })`
hidden.bs.tooltip	This event is fired when the tooltip has finished being hidden from the user *willwaitforCSStransitionstocomplete*.	`$('#myTooltip').on('hidden.bs.tooltip', function () { // do something… })`

Example

The following example demonstrates the use of tooltip plugin via data attributes.

```
<h4>Tooltip examples for anchors</h4>
This is a <a href="#"
   title="Default Tooltip">Default Tooltip
</a>.

<script>
$(function () { $('.tooltip-show').tooltip('show');});
$(function () { $('.tooltip-show').on('show.bs.tooltip', function () {
   alert("Alert message on show");
})});
</script>
```



BOOTSTRAP POPOVER PLUGIN

The popover is similar to tooltip, offering an extended view complete with a heading. For the popover to activate, a user just needs to hover the cursor over the element. The content of the popover can be populated entirely using the Bootstrap Data API. This method requires a tooltip.

> *If you want to include this plugin functionality individually, then you will need the **popover.js** and it has a dependency of the tooltip plugin. Else, as mentioned in the chapter Bootstrap Plugins Overview, you can include bootstrap.js or the minified bootstrap.min.js.*

Usage

The popover plugin generates content and markup on demand, and by default places popover

after their trigger element. You can add popover in the following two ways:

- **Via data attributes** : To add a popover, add **data-toggle="popover"** to an anchor/button tag. The title of the anchor will be the text of a popover. By default, popover is set to top by the plugin.

```
<a href="#" data-toggle="popover" title="Example popover">
   Hover over me
</a>
```

- **Via JavaScript** : Enable popovers via JavaScript using the following syntax:

```
$('#identifier').popover(options)
```

> *Popover plugin is NOT only-css plugins like dropdown or other plugins discussed in previous chapters. To use this plugin you MUST activate it using jquery readjavascript. To enable all the popovers on your page just use this script:*
> ```
> $(function () { $("[data-toggle='popover']").popover(); });
> ```

Example

The following example demonstrates the use of popover plugin via data attributes.

```
<div  >
   <button type="button"
      data-container="body" data-toggle="popover" data-placement="left"
      data-content="Some content in Popover on left">
      Popover on left
   </button>
   <button type="button"
      data-container="body" data-toggle="popover" data-placement="top"
      data-content="Some content in Popover on top">
      Popover on top
   </button>
   <button type="button"
      data-container="body" data-toggle="popover" data-placement="bottom"
      data-content="Some content in Popover on bottom">
      Popover on bottom
   </button>
   <button type="button"
      data-container="body" data-toggle="popover" data-placement="right"
      data-content="Some content in Popover on right">
      Popover on right
   </button>
   </div>

   <script>$(function ()
      { $("[data-toggle='popover']").popover();
      });
   </script>
</div>
```

Options

There are certain options which can be added via the Bootstrap Data API or invoked via JavaScript. Following table lists the options:

Option Name	Type/Default Value	Data attribute name	Description
animation	boolean *Default: true*	data-animation	Applies a CSS fade transition to the popover.
html	boolean *Default: false*	data-html	Inserts HTML into the popover. If false, jQuery's text method will be used to insert content into the dom. Use text if you're worried about XSS attacks.
placement	string|function *Default: top*	data-placement	Specifies how to position the popover *i. e. , top|bottom|left|right|auto*. When *auto* is specified, it will dynamically reorient the popover. For example, if placement is "auto left", the popover will display to the left when possible, otherwise it will display right.
selector	string *Default: false*	data-selector	If a selector is provided, popover objects will be delegated to the specified targets.
title	string | function *Default:* "	data-title	The title option is the default title value if the *title* attribute isn't present.
trigger	string *Default:* '*hover focus*'	data-trigger	Defines how the popover is triggered: **click| hover | focus | manual**. You may pass multiple triggers; separate them with a space.

delay	number | object
	Default: 0
		data-delay	Delays showing and hiding the popover in 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 *Default: false*	data-container	Appends the popover to a specific element. Example: container: 'body'

Methods

The following are some useful methods for popover:

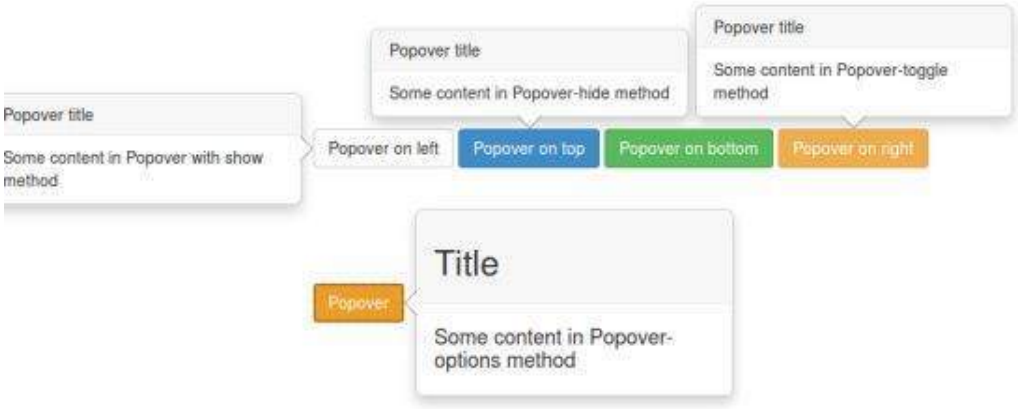
Method	Description	Example
Options: .popover*options*	Attaches a popover handler to an element collection.	`$().popover(options)`
Toggle: .popover *'toggle'*	Toggles an element's popover.	`$('#element').popover('toggle')`
Show: .popover *'show'*	Reveals an element's popover.	`$('#element').popover('show')`
Hide: .popover	Hides an element's popover.	`$('#element').popover('hide')`

'hide'		
Destroy: .popover*'destroy'*	Hides and destroys an element's popover.	`$('#element').popover('destroy')`

Example

The following example demonstrates the popover plugin methods:

```
<div  >
   <button type="button"
      title="Popover title" data-container="body"
      data-toggle="popover" data-placement="left"
      data-content="Some content in Popover with show method">
      Popover on left
   </button>
   <button type="button"
      title="Popover title" data-container="body"
      data-toggle="popover" data-placement="top"
      data-content="Some content in Popover-hide method">
      Popover on top
   </button>
   <button type="button"
      title="Popover title" data-container="body"
      data-toggle="popover" data-placement="bottom"
      data-content="Some content in Popover-destroy method">
      Popover on bottom
   </button>
   <button type="button"
      title="Popover title" data-container="body"
      data-toggle="popover" data-placement="top"
      data-content="Some content in Popover-toggle method">
      Popover on right
   </button><br><br><br><br><br><br>
   <p >
      <a href="#" type="button"
         data-container="body" data-toggle="popover" data-content="
         <h4>Some content in Popover-options method</h4>">
         Popover
      </a>
   </p>
   <script>
      $(function () { $('.popover-show').popover('show');});
      $(function () { $('.popover-hide').popover('hide');});
      $(function () { $('.popover-destroy').popover('destroy');});
      $(function () { $('.popover-toggle').popover('toggle');});
     $(function () { $(".popover-options a").popover({html : true });});
   </script>
</div>
```



Events

Following table lists the events to work with the popover plugin. This event may be used to hook into the function.

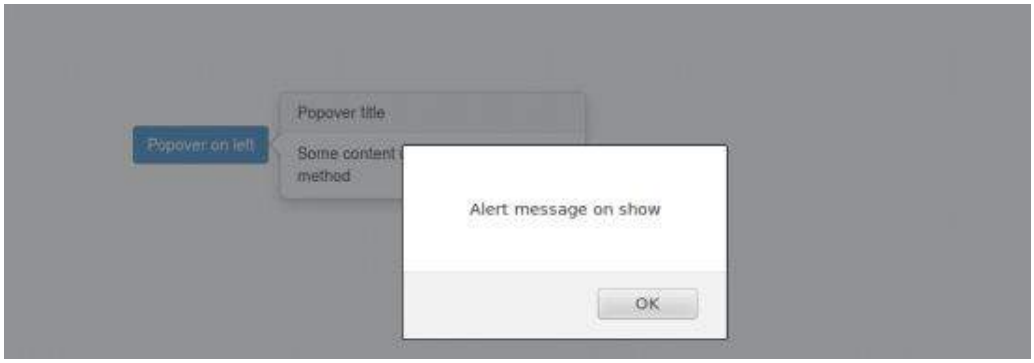
Event	Description	Example
show.bs.popover	This event fires immediately when the show instance method is called.	`$('#mypopover').on('show.bs.popover', function () { // do something… })`
shown.bs.popover	This event is fired when the popover has been made visible to the user *willwaitforCSStransitionstocomplete* ..	`$('#mypopover').on('shown.bs.popover', function () { // do something… })`
hide.bs.popover	This event is fired immediately when the hide instance method has been called.	`$('#mypopover').on('hide.bs.popover', function () { // do something… })`
hidden.bs.popover	This event is fired when the popover has finished being hidden from the user *willwaitforCSStransitionstocomplete* .	`$('#mypopover').on('hidden.bs.popover', function () { // do something… })`

Example

The following example demonstrates the Popover plugin events:

```
<div clas="container" style="padding: 100px 50px 10px;" >
   <button type="button"
      title="Popover title" data-container="body"
      data-toggle="popover"
      data-content="Some content in Popover with show method">
      Popover on left
   </button>

   </div>
   <script>
      $(function () { $('.popover-show').popover('show');});
      $(function () { $('.popover-show').on('shown.bs.popover', function () {
      alert("Alert message on show");
   })});
   </script>
</div>
```



BOOTSTRAP ALERT PLUGIN

Alert messages are mostly used to display information such as warning or confirmation messages to the end users. Using alert message plugin you can add dismiss functionality to all alert messages.

> *If you want to include this plugin functionality individually, then you will need the*

> ***alert.js**. Else, as mentioned in the chapter Bootstrap Plugins Overview, you can include the bootstrap.js or the minified bootstrap.min.js.*

Usage

You can enable dismissal of an alert in the following two ways:

- **Via data attributes** : To dismiss via Data API just add **data-dismiss="alert"** to your close button to automatically give an alert close functionality.

```
<a >
   &times;
</a>
```

- **Via JavaScript** : To dismiss via JavaScript use the following syntax:

```
$(".alert").alert()
```

Example

The following example demonstrates the use of alert plugin via data attributes.

```
<div >
   <a href="#" >
      &times;
   </a>
   <strong>Warning!</strong> There was a problem with your
   network connection.
</div>
```

Options

There are no options here.

Methods

The following are some useful methods for alert plugin:

Method	Description	Example
.alert	This method wraps all alerts with close functionality.	`$('#identifier').alert();`
Close Method .alert*'close'*	To enable all alerts to be closed, add this method.	`$('#identifier').alert('close');`

> *To enable alerts to animate out when closed, make sure they have the **.fade** and **.in** class already applied to them.*

Example

The following example demonstrates the use of **.alert** method:

```
<h3>Alert messages to demonstrate alert() method </h3>
<div  >
   <a href="#" >&times;</a>
```

```
   <strong>Success!</strong> the result is successful.
</div>
<div  >
   <a href="#" >&times;</a>
   <strong>Warning!</strong> There was a problem with your
   network connection.
</div>

<script type="text/javascript">
$(function(){
   $(".close").click(function(){
      $("#myAlert").alert();
   });
});
</script>
```

Try this code using the **Try it** editor.

The following example demonstrates the use of **.alert**'*close*' method:

```
<h3>Alert messages to demonstrate alert('close') method </h3>
<div  >
   <a href="#" >&times;</a>
   <strong>Success!</strong> the result is successful.
</div>
<div  >
   <a href="#" >&times;</a>
   <strong>Warning!</strong> There was a problem with your
   network connection.
</div>

<script type="text/javascript">
$(function(){
   $(".close").click(function(){
      $("#myAlert").alert('close');
   });
});

</script>
```

Try this code using the **Try it** editor. You can see that the close functionality is applied to all the alert messages i.e. close any alert message, rest of the alert messages also gets closed.

Events

Following table lists the events to work with alert plugin. This event may be used to hook into the alert function.

Event	Description	Example
close.bs.alert	This event fires immediately when the *close* instance method is called.	`$('#myalert').bind('close.bs.alert', function () {  // do something… })`
closed.bs.alert	This event is fired when the alert has been closed *willwaitforCSStransitionstocomplete*.	`$('#myalert').bind('closed.bs.alert', function () {  // do something… })`

Example

The following example demonstrates the alert plugin events:

```
<div  >
   <a href="#" >&times;</a>
```

```
   <strong>Success!</strong> the result is successful.
</div>

<script type="text/javascript">
$(function(){
   $("#myAlert").bind('closed.bs.alert', function () {
      alert("Alert message box is closed.");
   });
});
</script>
```

BOOTSTRAP BUTTON PLUGIN

Buttons were explained in chapter Bootstrap Buttons. With this plugin you can add in some interaction such as control button states or create groups of buttons for more components like toolbars.

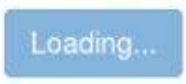
> *If you want to include this plugin functionality individually, then you will need the **button.js**. Else, as mentioned in the chapter Bootstrap Plugins Overview, you can include the bootstrap.js or the minified the bootstrap.min.js.*

Loading State

To add a loading state to a button, simply add the **data-loading-text="Loading..."** as an attribute to the button element as shown in the following example:

```
<button
   type="button"> Loading state
</button>
<script>
   $(function() {
      $(".btn").click(function(){
         $(this).button('loading').delay(1000).queue(function() {
           // $(this).button('reset');
         });
      });
   });
</script>
```

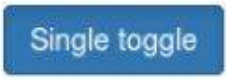
When you click on the button the output would be as seen in the following image:



Single toggle

To activate toggling *i. e. changethenormalstateofabuttontoapushstateandviceversa* on a single button, add the **data-toggle="button"** as an attribute to the button element as shown in the following example:

```
<button type="button"
   data-toggle="button">Single toggle
</button>
```



Checkbox

You can create group of checkboxes and add toggling to it by simply adding the data attribute **data-toggle="buttons"** to the **btn-group.**

```
<div >
   <label >
      <input type="checkbox"> Option 1
   </label>
   <label >
      <input type="checkbox"> Option 2
   </label>
   <label >
      <input type="checkbox"> Option 3
   </label>
</div>
```

Radio

Similarly you can create a group of radio inputs and add toggling to it by simply adding the data attribute **data-toggle="buttons"** to the **btn-group**.

```
<div >
   <label >
      <input type="radio" name="options" > Option 1
   </label>
   <label >
      <input type="radio" name="options" > Option 2
   </label>
   <label >
      <input type="radio" name="options" > Option 3
   </label>
</div>
```

Usage

You can enable buttons plugin **via JavaScript** as shown below:

```
$('.btn').button()
```

Options

There are no options.

Methods

Given below are some of the useful methods for buttons plugin:

Method	Description	Example
button *'toggle'*	Toggles push state. Gives the button the appearance that it has been activated. You can also enable auto toggling of a button by using the **data-toggle** attribute.	`$().button('toggle')`
.button *'loading'*	When loading, the button is disabled and the text is changed to the option from the **data-loading-text** attribute of button element	`$().button('loading')`
.button *'reset'*	Resets button state, bringing the original content back to the text. This method is useful when you need to return the button back to the primary state	`$().button('reset')`
.button *string*	String in this method is referring to any string declared by the user.To reset the button state and bring in new content use this method.	`$().button(string)`

Example

The following example demonstrates the use of the above methods:

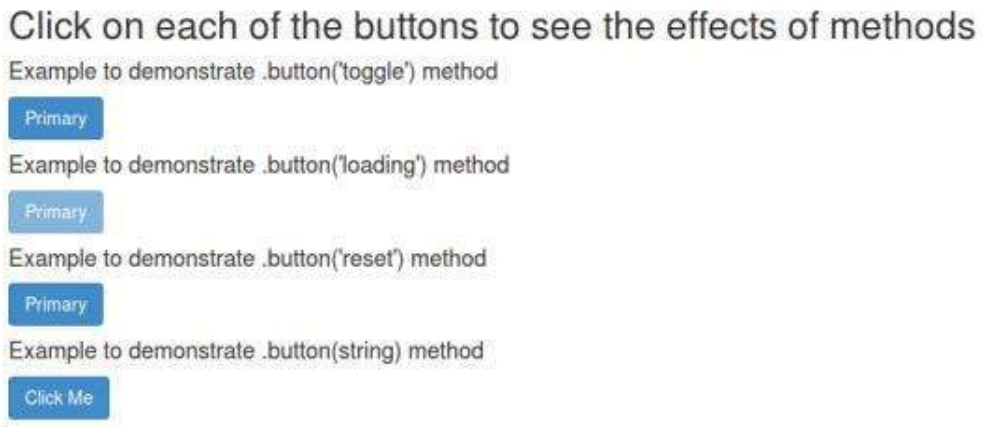
```
<h2>Click on each of the buttons to see the effects of methods</h2>
<h4>Example to demonstrate  .button('toggle') method</h4>
<div  >
   <button type="button" >Primary</button>
</div>

<h4>Example to demonstrate  .button('loading') method</h4>
<div  >
   <button type="button"
      data-loading-text="Loading...">Primary
   </button>
</div>

<h4>Example to demonstrate  .button('reset') method</h4>
<div  >
   <button type="button"
      data-loading-text="Loading...">Primary
   </button>
</div>

<h4>Example to demonstrate  .button(string) method</h4>
<button type="button"
   data-complete-text="Loading finished">Click Me
</button>
<script type="text/javascript">
   $(function () {
      $("#myButtons1 .btn").click(function(){
         $(this).button('toggle');
      });
   });
   $(function() {
      $("#myButtons2 .btn").click(function(){
         $(this).button('loading').delay(1000).queue(function() {
         });
      });
   });
   $(function() {
      $("#myButtons3 .btn").click(function(){
         $(this).button('loading').delay(1000).queue(function() {
            $(this).button('reset');
         });
      });
   });
   $(function() {
      $("#myButton4").click(function(){
         $(this).button('loading').delay(1000).queue(function() {
```

```
            $(this).button('complete');
        });
    });
});
</script>
```



BOOTSTRAP COLLAPSE PLUGIN

The collapse plugin makes it easy to make collapsing divisions of the page. Whether you use it to build an accordion navigation or content boxes, it allows for a lot of content options.

> *If you want to include this plugin functionality individually, then you will need the **collapse.js**. This also requires the Transition Plugin to be included in your version of Bootstrap. Else, as mentioned in the chapter Bootstrap Plugins Overview, you can include thebootstrap.js or the minified bootstrap.min.js.*

You can use the collapse plugin:

- ***To create collapsible groups or accordion***. This can be created as in the sample example below:

```
<div >
  <div >
    <div >
      <h4 >
        <a data-toggle="collapse" data-parent="#accordion"
          href="#collapseOne">
          Click me to exapand. Click me again to collapse.Section 1
        </a>
      </h4>
    </div>
    <div  >
      <div >
        Nihil anim keffiyeh helvetica, craft beer labore wes anderson
        cred nesciunt sapiente ea proident. Ad vegan excepteur butcher
        vice lomo.
      </div>
    </div>
  </div>
  <div >
    <div >
      <h4 >
        <a data-toggle="collapse" data-parent="#accordion"
          href="#collapseTwo">
          Click me to exapand. Click me again to collapse.Section 2
        </a>
      </h4>
    </div>
    <div  >
      <div >
        Nihil anim keffiyeh helvetica, craft beer labore wes anderson
```

```
        cred nesciunt sapiente ea proident. Ad vegan excepteur butcher
        vice lomo.
      </div>
    </div>
  </div>
  <div >
    <div >
      <h4 >
        <a data-toggle="collapse" data-parent="#accordion"
          href="#collapseThree">
          Click me to exapand. Click me again to collapse.Section 3
        </a>
      </h4>
    </div>
    <div  >
      <div >
        Nihil anim keffiyeh helvetica, craft beer labore wes anderson
        cred nesciunt sapiente ea proident. Ad vegan excepteur butcher
        vice lomo.
      </div>
    </div>
  </div>
</div>
```

- **data-toggle="collapse"** is added to the link on which you click to expand or collapse the component.
- **href** or a **data-target** attribute is added to the parent component, whose value is *id* of the child component.
- **data-parent** attribute is added for creating the accordion like effect.

- ***To create simple collapsible without the accordion markup***: This can be created as in the sample example below:

```
<button type="button"
   data-target="#demo">
   simple collapsible
</button>

<div  >
  Nihil anim keffiyeh helvetica, craft beer labore wes anderson
  cred nesciunt sapiente ea proident. Ad vegan excepteur butcher
  vice lomo.
</div>
```

As you can see in the example we have created a simple collapsible component, unlike accordion, we haven't added the attribute **data-parent**.

Usage

Following table lists the classes that the collapse plugin utilizes to handle the heavy lifting:

Class	Description

.collapse	Hides the content.
.collapse.in	Shows the content.
.collapsing	Is added when the transition starts, and removed when it finishes.

You can use collapse plugin in two ways:

- **Via data attributes** : Add **data-toggle="collapse"** and a **data-target** to the element to automatically assign control of a collapsible element. The **data-target** attribute will accept a CSS selector to apply the collapse to. Be sure to add the class **.collapse** to the collapsible element. If you'd like it to default open, include the additional class **.in**.

  To add accordion-like group management to a collapsible control, add the data attribute **data-parent="#selector"**.

- **Via JavaScript**: The collapse method can be activated with JavaScript as shown below:

```
$('.collapse').collapse()
```

Options

There are certain options which can be passed via data attributes or JavaScript are listed in the following table:

Option Name	Type/Default Value	Data attribute name	Description
parent	selector *Default: false*	data-parent	If selector is false, then all collapsible elements under the specified parent will be closed *similartotraditionalaccordionbehavior − thisdependentontheaccordion − groupclass* .
toggle	boolean *Default: true*	data-toggle	Toggles the collapsible element on invocation.

Methods

Here is a list of some useful methods that are used with collapsible elements.

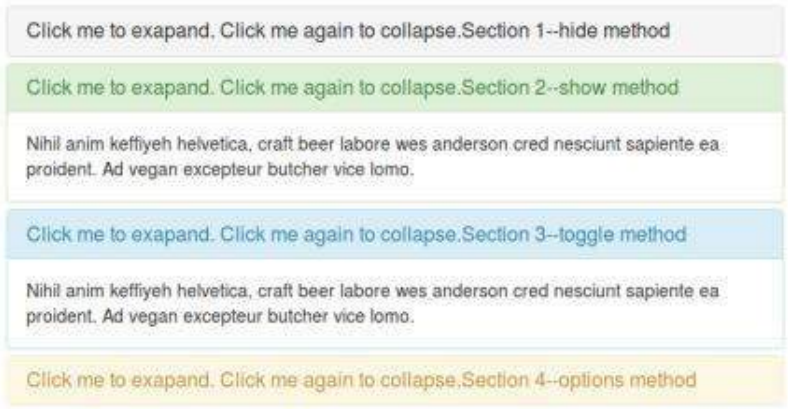
Method	Description	Example
Options: .collapse *options*	Activates your content as a collapsible element. Accepts an optional options object.	`$('#identifier').collapse({ toggle: false })`
Toggle: .collapse *'toggle'*	Toggles a collapsible element to shown or hidden.	`$('#identifier').collapse('toggle')`
Show: .collapse *'show'*	Shows a collapsible element.	`$('#identifier').collapse('show')`
Hide: .collapse *'hide'*	Hides a collapsible element.	`$('#identifier').collapse('hide')`

Example

The following example demonstrates the usage of methods:

```
<div >
   <div >
      <div >
         <h4 >
            <a data-toggle="collapse" data-parent="#accordion"
               href="#collapseOne">
               Click me to exapand. Click me again to collapse.
               Section 1--hide method
            </a>
         </h4>
      </div>
      <div  >
         <div >
            Nihil anim keffiyeh helvetica, craft beer labore wes anderson cred
            nesciunt sapiente ea proident. Ad vegan excepteur butcher vice
            lomo.
         </div>
      </div>
   </div>
   <div >
      <div >
         <h4 >
            <a data-toggle="collapse" data-parent="#accordion"
               href="#collapseTwo">
               Click me to exapand. Click me again to collapse.
               Section 2--show method
            </a>
         </h4>
      </div>
      <div  >
         <div >
            Nihil anim keffiyeh helvetica, craft beer labore wes anderson cred
            nesciunt sapiente ea proident. Ad vegan excepteur butcher vice
            lomo.
         </div>
      </div>
   </div>
   <div >
      <div >
         <h4 >
            <a data-toggle="collapse" data-parent="#accordion"
               href="#collapseThree">
               Click me to exapand. Click me again to collapse.
               Section 3--toggle method
            </a>
         </h4>
      </div>
      <div  >
         <div >
            Nihil anim keffiyeh helvetica, craft beer labore wes anderson cred
            nesciunt sapiente ea proident. Ad vegan excepteur butcher vice
            lomo.
         </div>
      </div>
   </div>
   <div >
      <div >
         <h4 >
            <a data-toggle="collapse" data-parent="#accordion"
               href="#collapseFour">
               Click me to exapand. Click me again to collapse.
               Section 4--options method
            </a>
         </h4>
      </div>
      <div  >
         <div >
            Nihil anim keffiyeh helvetica, craft beer labore wes anderson cred
            nesciunt sapiente ea proident. Ad vegan excepteur butcher vice
            lomo.
         </div>
      </div>
```

```
   </div>
</div>
<script type="text/javascript">
   $(function () { $('#collapseFour').collapse({
      toggle: false
   })});
   $(function () { $('#collapseTwo').collapse('show')});
   $(function () { $('#collapseThree').collapse('toggle')});
   $(function () { $('#collapseOne').collapse('hide')});
</script>
```



Events

The following table lists a few events that can be used with the collapse functionality.

Event	Description	Example
show.bs.collapse	Fired after the show method is called.	`$('#identifier').on('show.bs.collapse', function () { // do something… })`
shown.bs.collapse	This event is fired when a collapse element has been made visible to the user *willwaitforCSStransitionstocomplete* .	`$('#identifier').on('shown.bs.collapse', function () { // do something… })`
hide.bs.collapse	Fired when the hide instance method has been called.	`$('#identifier').on('hide.bs.collapse', function () { // do something… })`
hidden.bs.collapse	This event is fired when a collapse element has been hidden from the user *willwaitforCSStransitionstocomplete* .	`$('#identifier').on('hidden.bs.collapse', function () { // do something… })`

Example

The following example demonstrates the usage of events:

```
<div >
   <div >
      <div >
         <h4 >
            <a data-toggle="collapse" data-parent="#accordion"
               href="#collapseexample">
               Click me to exapand. Click me again to collapse.
               Section --shown event
```

```
            </a>
         </h4>
      </div>
      <div  >
         <div >
            Nihil anim keffiyeh helvetica, craft beer labore wes anderson
            cred nesciunt sapiente ea proident.
            Ad vegan excepteur butcher vice lomo.
         </div>
      </div>
   </div>
</div>

<script type="text/javascript">
   $(function () {
      $('#collapseexample').on('show.bs.collapse', function () {
         alert('Hey, this alert shows up when you expand it');})
   });
</script>
```

BOOTSTRAP CAROUSEL PLUGIN

The Bootstrap carousel is a flexible, responsive way to add a slider to your site. In addition to being responsive, the content is flexible enough to allow images, iframes, videos, or just about any type of content that you might want.

> *If you want to include this plugin functionality individually, then you will need the **carousel.js**. Else, as mentioned in the chapter Bootstrap Plugins Overview, you can include the bootstrap.js or the minified bootstrap.min.js.*

Example

A simple slideshow below shows a generic component for cycling through the elements like a carousel, using the Bootstrap carousel plugin. To implement the carousel, you just need to add the code with the markup. There is no need for data attributes, just simple class-based development.

```
<div  >
   <!-- Carousel indicators -->
   <ol >
      <li data-target="#myCarousel" data-slide-to="0" ></li>
      <li data-target="#myCarousel" data-slide-to="1"></li>
      <li data-target="#myCarousel" data-slide-to="2"></li>
   </ol>
   <!-- Carousel items -->
   <div >
      <div >
         <img src="/bootstrap/images/slide1.png" alt="First slide">
      </div>
      <div >
         <img src="/bootstrap/images/slide2.png" alt="Second slide">
      </div>
      <div >
         <img src="/bootstrap/images/slide3.png" alt="Third slide">
      </div>
   </div>
   <!-- Carousel nav -->
   <a
      data-slide="prev">&lsaquo;</a>
```

```
   <a
      data-slide="next">&rsaquo;</a>
</div>
```

Optional Captions

You can add captions to your slides easily with the **.carousel-caption** element within any **.item**. Place just about any optional HTML within there and it will be automatically aligned and formatted. The following example demonstrates this:

```
<div  >
   <!-- Carousel indicators -->
   <ol >
      <li data-target="#myCarousel" data-slide-to="0" ></li>
      <li data-target="#myCarousel" data-slide-to="1"></li>
      <li data-target="#myCarousel" data-slide-to="2"></li>
   </ol>
   <!-- Carousel items -->
   <div >
      <div >
         <img src="/bootstrap/images/slide1.png" alt="First slide">
         <div >This Caption 1</div>
      </div>
      <div >
         <img src="/bootstrap/images/slide2.png" alt="Second slide">
         <div >This Caption 2</div>
      </div>
      <div >
         <img src="/bootstrap/images/slide3.png" alt="Third slide">
         <div >This Caption 3</div>
      </div>
   </div>
   <!-- Carousel nav -->
   <a
      data-slide="prev">&lsaquo;</a>
   <a
      data-slide="next">&rsaquo;</a>
</div>
```

Usage

- **Via data attributes** : Use data attributes to easily control the position of the carousel.
    - Attribute **data-slide** accepts the keywords *prev* or *next*, which alters the slide position relative to its current position.
    - Use **data-slide-to** to pass a raw slide index to the carousel **data-slide-to="2"**, which shifts the slide position to a particular index beginning with 0.
    - The **data-ride="carousel"** attribute is used to mark a carousel as an animation starting at page load.
- **Via JavaScript** : The carousel can be manually called with JavaScript as below:

```
$('.carousel').carousel()
```

Options

There are certain, options which can be passed via data attributes or JavaScript are listed in the following table:

Option Name	Type/Default Value	Data attribute name	Description
interval	number *Default: 5000*	data-interval	The amount of time to delay between automatically cycling an item. If false, carousel will not automatically cycle.
pause	string *Default:* "*hover*"	data-pause	Pauses the cycling of the carousel on mouseenter and resumes the cycling of the carousel on mouseleave.
wrap	boolean *Default: true*	data-wrap	Whether the carousel should cycle continuously or have hard stops.

Methods

Here is a list of useful methods that can be used with carousel code.

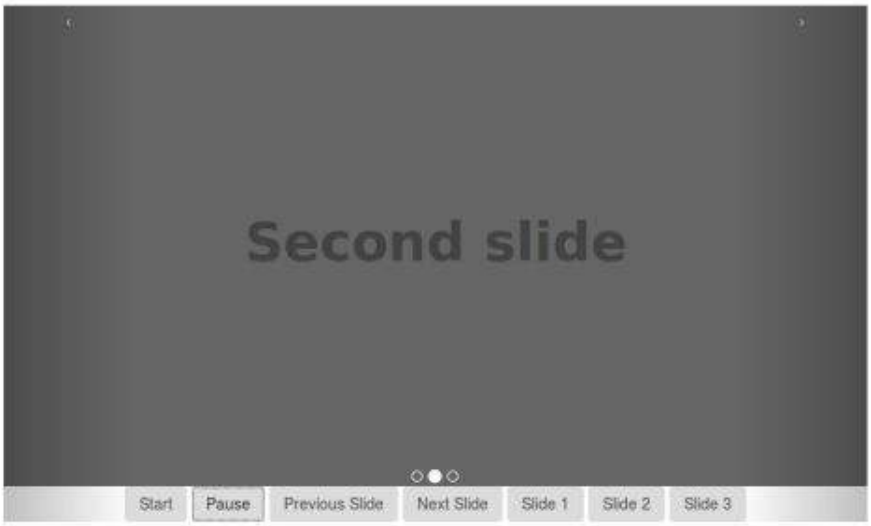
Method	Description	Example
.carousel *options*	Initializes the carousel with an optional options object and starts cycling through items.	`$('#identifier').carousel({ interval: 2000 })`
.carousel *'cycle'*	Cycles through the carousel items from left to right.	`$('#identifier').carousel('cycle')`
.carousel *'pause'*	Stops the carousel from cycling through items.	`$('#identifier')..carousel('pause')`
.carousel *number*	Cycles the carousel to a particular frame *0based, similartoanarray*.	`$('#identifier').carousel(number)`
.carousel *'prev'*	Cycles to the previous item.	`$('#identifier').carousel('prev')`
.carousel	Cycles to the next item.	`$('#identifier').carousel('next')`

'next'

Example

The following example demonstrates the usage of methods:

```
<div  >
   <!-- Carousel indicators -->
   <ol >
      <li data-target="#myCarousel" data-slide-to="0"
         ></li>
      <li data-target="#myCarousel" data-slide-to="1"></li>
      <li data-target="#myCarousel" data-slide-to="2"></li>
   </ol>
   <!-- Carousel items -->
   <div >
      <div >
         <img src="/bootstrap/images/slide1.png" alt="First slide">
      </div>
      <div >
         <img src="/bootstrap/images/slide2.png" alt="Second slide">
      </div>
      <div >
         <img src="/bootstrap/images/slide3.png" alt="Third slide">
      </div>
   </div>
   <!-- Carousel nav -->
   <a
      data-slide="prev">&lsaquo;</a>
   <a
      data-slide="next">&rsaquo;</a>
   <!-- Controls buttons -->
   <div style="text-align:center;">
      <input type="button" >
      <input type="button" >
      <input type="button" >
      <input type="button" >
      <input type="button" >
      <input type="button" >
      <input type="button" >
   </div>
</div>
<script>
   $(function(){
      // Initializes the carousel
      $(".start-slide").click(function(){
         $("#myCarousel").carousel('cycle');
      });
      // Stops the carousel
      $(".pause-slide").click(function(){
         $("#myCarousel").carousel('pause');
      });
      // Cycles to the previous item
      $(".prev-slide").click(function(){
         $("#myCarousel").carousel('prev');
      });
      // Cycles to the next item
      $(".next-slide").click(function(){
         $("#myCarousel").carousel('next');
      });
      // Cycles the carousel to a particular frame
      $(".slide-one").click(function(){
         $("#myCarousel").carousel(0);
      });
      $(".slide-two").click(function(){
         $("#myCarousel").carousel(1);
      });
      $(".slide-three").click(function(){
         $("#myCarousel").carousel(2);
      });
   });
</script>
```



Events

Bootstrap's carousel class exposes two events for hooking into carousel functionality which are listed in the following table.

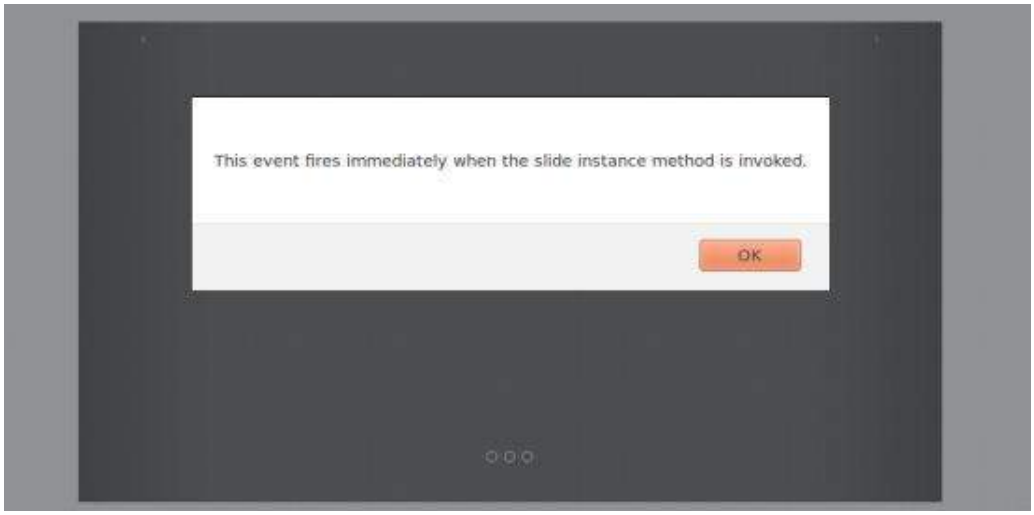
Event	Description	Example
slide.bs.carousel	This event fires immediately when the slide instance method is invoked..	`$('#identifier').on('slide.bs.carousel', function () { // do something… })`
slid.bs.carousel	This event is fired when the carousel has completed its slide transition.	`$('#identifier').on('slid.bs.carousel', function () { // do something… })`

Example

The following example demonstrates the usage of events:

```
<div  >
   <!-- Carousel indicators -->
   <ol >
      <li data-target="#myCarousel" data-slide-to="0"
         ></li>
      <li data-target="#myCarousel" data-slide-to="1"></li>
      <li data-target="#myCarousel" data-slide-to="2"></li>
   </ol>
   <!-- Carousel items -->
   <div >
      <div >
         <img src="/bootstrap/images/slide1.png" alt="First slide">
      </div>
      <div >
         <img src="/bootstrap/images/slide2.png" alt="Second slide">
      </div>
      <div >
         <img src="/bootstrap/images/slide3.png" alt="Third slide">
      </div>
   </div>
   <!-- Carousel nav -->
   <a
      data-slide="prev">&lsaquo;</a>
   <a
      data-slide="next">&rsaquo;</a>
</div>
```

```
<script>
   $(function(){
      $('#myCarousel').on('slide.bs.carousel', function () {
         alert("This event fires immediately when the slide instance method"
            +"is invoked.");
      });
   });
</script>
```



BOOTSTRAP AFFIX PLUGIN

The affix plugin allows a to become affixed to a location on the page. You can also toggle it's pinning on and off using this plugin. A common example of this are social icons. They will start in a location, but as the page hits a certain mark, the will be locked in place and will stop scrolling with the rest of the page.

> *If you want to include this plugin functionality individually, then you will need the **affix.js**. Else, as mentioned in the chapter Bootstrap Plugins Overview, you can include the bootstrap.js or the minified bootstrap.min.js.*

Usage

You can use the affix plugin via data attributes or manually with your own JavaScript as discussed below.

- **Via data attributes** : To easily add affix behavior to any element, just add **data-spy="affix"** to the element you want to spy on. Use offsets to define when to toggle the pinning of an element.

Example

The following example demonstrates the usage through data attributes:

```
<div >
   <div >
      <h1>Bootstrap Affix Plugin example</h1>
   </div>
   <div
      data-offset-bottom="200">
      <div >
         <ul
            data-offset-top="190">
            <li >Tutorial One</a></li>
            <li><a href="#two">Tutorial Two</a></li>
            <li><a href="#three">Tutorial Three</a></li>
         </ul>
      </div>
      <div >
         <h2 >Tutorial One</h2>
         <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.
         Nam eu sem tempor, varius quam at, luctus dui. Mauris magna
```

```
        metus, dapibus nec turpis vel, semper malesuada ante.
        Vestibulum id metus ac nisl bibendum scelerisque non non
        purus. Suspendisse varius nibh non aliquet sagittis. In
        tincidunt orci sit amet elementum vestibulum. Vivamus
        fermentum in arcu in aliquam. Quisque aliquam porta odio
        in fringilla. Vivamus nisl leo, blandit at bibendum eu,
        tristique eget risus. Integer aliquet quam ut elit suscipit,
        id interdum neque porttitor. Integer
        faucibus ligula.</p>
        <p>Vestibulum quis quam ut magna consequat faucibus.
        Pellentesque eget nisi a mi suscipit tincidunt. Ut tempus
        dictum risus. Pellentesque viverra sagittis quam at mattis.
        Suspendisse potenti. Aliquam sit amet gravida nibh,
        facilisis gravida odio. Phasellus auctor velit at lacus
        blandit, commodo iaculis justo  viverra. Etiam vitae est
        arcu. Mauris vel congue dolor. Aliquam eget mi mi. Fusce
        quam tortor, commodo ac dui quis, bibendum viverra erat.
        Maecenas mattis lectus enim, quis tincidunt dui molestie
        euismod. Curabitur et diam tristique, accumsan nunc eu,
        hendrerit tellus.</p>
        <hr>
        <h2 >Tutorial Two</h2>
        <p> Nullam hendrerit justo non leo aliquet imperdiet. Etiam
        in sagittis lectus. Suspendisse ultrices placerat accumsan.
        Mauris quis dapibus orci. In dapibus velit blandit pharetra
        tincidunt. Quisque non sapien nec lacus condimentum facilisis
        ut iaculis enim. Sed viverra interdum bibendum. Donec ac
        sollicitudin dolor. Sed fringilla vitae lacus at rutrum.
        Phasellus congue vestibulum ligula sed consequat.</p>
        <p>Vestibulum consectetur scelerisque lacus, ac fermentum
        lorem convallis sed. Nam odio tortor, dictum quis malesuada at,
        pellentesque vitae orci. Vivamus elementum, felis eu auctor
        lobortis, diam velit egestas lacus, quis fermentum metus ante
        quis urna. Sed at facilisis libero. Cum sociis natoque
        penatibus et magnis dis parturient montes, nascetur ridiculus
        mus. Vestibulum bibendum blandit dolor. Nunc orci dolor,
        molestie nec nibh in, hendrerit tincidunt ante. Vivamus sem
        augue, hendrerit non sapien in, mollis ornare augue.</p>
        <hr>
        <h2 >Tutorial Three</h2>
        <p>Integer pulvinar leo id risus pellentesque vestibulum.
        Sed diam libero, sodales eget sapien vel, porttitor bibendum
        enim. Donec sed nibh vitae lorem porttitor blandit in nec ante.
        Pellentesque vitae metus ipsum. Phasellus sed nunc ac sem
        malesuada condimentum. Etiam in aliquam lectus. Nam vel sapien
        diam. Donec pharetra id arcu eget blandit. Proin imperdiet
        mattis augue in porttitor. Quisque tempus enim id lobortis
        feugiat. Suspendisse tincidunt risus quis dolor fringilla
        blandit. Ut sed sapien at purus lacinia porttitor. Nullam
        iaculis, felis a pretium ornare, dolor nisl semper tortor, vel
        sagittis lacus est consequat eros. Sed id pretium nisl.
        Curabitur dolor nisl, laoreet vitae aliquam id, tincidunt sit
        amet mauris. </p>
        <p>Phasellus vitae suscipit justo. Mauris pharetra feugiat
        ante id lacinia. Etiam faucibus mauris id tempor egestas. Duis
        luctus turpis at accumsan tincidunt. Phasellus risus risus,
        volutpat vel tellus ac, tincidunt fringilla massa. Etiam
        hendrerit dolor eget ante rutrum adipiscing. Cras interdum
        ipsum mattis, tempus mauris vel, semper ipsum. Duis sed dolor
        ut enim lobortis pellentesque ultricies ac ligula. Pellentesque
        convallis  elit nisi, id vulputate ipsum ullamcorper ut. Cras
        ac pulvinar purus, ac viverra est. Suspendisse potenti. Integer
        pellentesque neque et elementum tempus. Curabitur bibendum in
        ligula ut rhoncus.</p>
        <p>Quisque pharetra velit id velit iaculis pretium. Nullam a
        justo sed ligula porta semper eu quis enim. Pellentesque
        pellentesque, metus at facilisis hendrerit, lectus velit
        facilisis leo, quis volutpat turpis arcu quis enim. Nulla
        viverra lorem elementum interdum ultricies. Suspendisse
        accumsan quam nec ante mollis tempus. Morbi vel accumsan diam,
        eget convallis tellus. Suspendisse potenti.</p>
      </div>
   </div>
</div>
```

- **Via JavaScript** : You can affix an element manually with JavaScript as shown below:

```
$('#myAffix').affix({
   offset: {
      top: 100, bottom: function () {
         return (this.bottom =
            $('.bs-footer').outerHeight(true))
         }
      }
})
```

Example

The following example demonstrates the usage through data attributes:

```
<div >
   <div >
      <h1>Bootstrap Affix Plugin example</h1>
   </div>
   <div>
      <div >
         <ul >
            <li >Tutorial One</a></li>
            <li><a href="#two">Tutorial Two</a></li>
            <li><a href="#three">Tutorial Three</a></li>
         </ul>
      </div>
      <div >
         <h2 >Tutorial One</h2>
         <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.
         Nam eu sem tempor, varius quam at, luctus dui. Mauris magna
         metus, dapibus nec turpis vel, semper malesuada ante.
         Vestibulum id metus ac nisl bibendum scelerisque non non
         purus. Suspendisse varius nibh non aliquet sagittis. In
         tincidunt orci sit amet elementum vestibulum. Vivamus
         fermentum in arcu in aliquam. Quisque aliquam porta odio
         in fringilla. Vivamus nisl leo, blandit at bibendum eu,
         tristique eget risus. Integer aliquet quam ut elit suscipit,
         id interdum neque porttitor. Integer
         faucibus ligula.</p>
         <p>Vestibulum quis quam ut magna consequat faucibus.
         Pellentesque eget nisi a mi suscipit tincidunt. Ut tempus
         dictum risus. Pellentesque viverra sagittis quam at mattis.
         Suspendisse potenti. Aliquam sit amet gravida nibh,
         facilisis gravida odio. Phasellus auctor velit at lacus
         blandit, commodo iaculis justo  viverra. Etiam vitae est
         arcu. Mauris vel congue dolor. Aliquam eget mi mi. Fusce
         quam tortor, commodo ac dui quis, bibendum viverra erat.
         Maecenas mattis lectus enim, quis tincidunt dui molestie
         euismod. Curabitur et diam tristique, accumsan nunc eu,
         hendrerit tellus.</p>
         <hr>
```

```
         <h2 >Tutorial Two</h2>
         <p> Nullam hendrerit justo non leo aliquet imperdiet. Etiam
         in sagittis lectus. Suspendisse ultrices placerat accumsan.
         Mauris quis dapibus orci. In dapibus velit blandit pharetra
         tincidunt. Quisque non sapien nec lacus condimentum facilisis
         ut iaculis enim. Sed viverra interdum bibendum. Donec ac
         sollicitudin dolor. Sed fringilla vitae lacus at rutrum.
         Phasellus congue vestibulum ligula sed consequat.</p>
         <p>Vestibulum consectetur scelerisque lacus, ac fermentum
         lorem convallis sed. Nam odio tortor, dictum quis malesuada at,
         pellentesque vitae orci. Vivamus elementum, felis eu auctor
         lobortis, diam velit egestas lacus, quis fermentum metus ante
         quis urna. Sed at facilisis libero. Cum sociis natoque
         penatibus et magnis dis parturient montes, nascetur ridiculus
         mus. Vestibulum bibendum blandit dolor. Nunc orci dolor,
         molestie nec nibh in, hendrerit tincidunt ante. Vivamus sem
         augue, hendrerit non sapien in, mollis ornare augue.</p>
         <hr>
         <h2 >Tutorial Three</h2>
         <p>Integer pulvinar leo id risus pellentesque vestibulum.
         Sed diam libero, sodales eget sapien vel, porttitor bibendum
         enim. Donec sed nibh vitae lorem porttitor blandit in nec ante.
         Pellentesque vitae metus ipsum. Phasellus sed nunc ac sem
         malesuada condimentum. Etiam in aliquam lectus. Nam vel sapien
         diam. Donec pharetra id arcu eget blandit. Proin imperdiet
         mattis augue in porttitor. Quisque tempus enim id lobortis
         feugiat. Suspendisse tincidunt risus quis dolor fringilla
         blandit. Ut sed sapien at purus lacinia porttitor. Nullam
         iaculis, felis a pretium ornare, dolor nisl semper tortor, vel
         sagittis lacus est consequat eros. Sed id pretium nisl.
         Curabitur dolor nisl, laoreet vitae aliquam id, tincidunt sit
         amet mauris. </p>
         <p>Phasellus vitae suscipit justo. Mauris pharetra feugiat
         ante id lacinia. Etiam faucibus mauris id tempor egestas. Duis
         luctus turpis at accumsan tincidunt. Phasellus risus risus,
         volutpat vel tellus ac, tincidunt fringilla massa. Etiam
         hendrerit dolor eget ante rutrum adipiscing. Cras interdum
         ipsum mattis, tempus mauris vel, semper ipsum. Duis sed dolor
         ut enim lobortis pellentesque ultricies ac ligula. Pellentesque
         convallis  elit nisi, id vulputate ipsum ullamcorper ut. Cras
         ac pulvinar purus, ac viverra est. Suspendisse potenti. Integer
         pellentesque neque et elementum tempus. Curabitur bibendum in
         ligula ut rhoncus.</p>
         <p>Quisque pharetra velit id velit iaculis pretium. Nullam a
         justo sed ligula porta semper eu quis enim. Pellentesque
         pellentesque, metus at facilisis hendrerit, lectus velit
         facilisis leo, quis volutpat turpis arcu quis enim. Nulla
         viverra lorem elementum interdum ultricies. Suspendisse
         accumsan quam nec ante mollis tempus. Morbi vel accumsan diam,
         eget convallis tellus. Suspendisse potenti.</p>
      </div>
   </div>
</div>
<script type="text/javascript">
   $(function () {
      $('#myNav').affix({
         offset: {
            top: 60
         }
      });
   });
</script>
```

Positioning via CSS

In both the above situations, you must provide CSS for the positioning of your content. The affix plugin toggles between three classes, each representing a particular state: *.affix, .affix-top, and .affix-bottom*. Follow the below steps to set your CSS for either of the above usage options.

- To start, the plugin adds **.affix-top** to indicate the element is in its top-most position. At this point no CSS positioning is required.
- Scrolling past the element you want affixed should trigger the actual affixing. This is where **.affix** replaces **.affix-top** and sets **position: fixed;** *providedbyBootstrap*′*scodeCSS*.
- If a bottom offset is defined, scrolling past that should replace **.affix** with **.affix-bottom**. Since offsets are optional, setting one requires you to set the appropriate CSS. In this case, add **position: absolute;** when necessary.

Options

There are certain options which can be passed via data attributes or JavaScript as listed in the following table:

Option Name	Type/Default Value	Data attribute name	Description
offset	number | function | object *Default: 10*	data-offset	Pixels to offset from screen when calculating position of scroll. If a single number is provided, the offset will be applied in both the top and bottom directions. To provide a unique, bottom and top offset just provide an object offset: { top: 10 } or offset: { top: 10, bottom: 5 }. Use a function when you need to dynamically calculate an offset.