

Responsive Design

Design that Adapts to
Different Devices



Responsive Design

Overview

Table of Contents

- Responsive Design
- Creating Responsive Design
 - Fluid Layout
 - Flexible Images and Fonts
 - Flexible Tables and Menus
 - Media Queries
- Media Queries Everywhere
- Constructing Responsive Design



2

Responsive Design

- Responsive design (adaptive design) is an approach to optimize the viewing experience on range of devices
 - Better user experience on mobile, desktop, TV
 - The UI depends on the device and device specifics
 - Screen size
 - Screen resolution
 - DPI
 - Color range
 - Etc...



4

Responsive Design

Live Demo



Creating Responsive Design

■ Responsive design involves one or more of the following:

- Fluid layout
 - Use proportional values for widths, margins, etc...
- Flexible fonts
 - Fonts based on the root
- Flexible images
 - Images cannot go beyond their container
- Media queries
 - Apply styles based on the client screen size

7

Creating Responsive Design

Ways to Implement a Nice UI



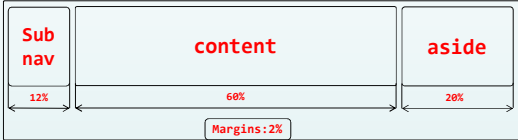
Fluid Layout

Make the Elements Flow on the Screen



Fluid Layout

- Fluid layout uses proportional sizes




The diagram shows a horizontal container divided into three sections. The first section on the left is labeled 'Sub nav' and has a width of 12%. The middle section is labeled 'content' and has a width of 60%. The third section on the right is labeled 'aside' and has a width of 20%. Below the sections, a label indicates 'Margins: 2%'.

- Pros:**
 - UI responds better to the client resolution
 - Saves code on media queries
- Cons:**
 - More whitespace on large screens (TV)

9


Flexible Images



A photograph showing a person's hands holding a tablet. The tablet screen displays a contact list application with a grid of contact cards. The image demonstrates how the layout of the application adapts to the physical dimensions of the device.

Fluid Layout

Live Demo



A screenshot of a live demo. It features a large rectangular container filled with a blue, abstract, wavy pattern. The container is centered on a light blue background.

Flexible Images

- Fluid design gets broken when using elements with fixed size
 - By concept images are always with fixed size
- Example:
 - Resolution: 1024px, container with width: 60% ($= 60\% * 1024 = 614.4\text{px}$) and an image with width: 500px
 - Seems OK
 - When the resolution becomes 780px, the container's width is still 60% ($= 468\text{px}$), and the image's width is still 500px
 - The image overflows its container

12

Slide 9

A1

Author, 12/27/2016

Making Images Flexible

- The fix to the image overflow is simple
 - Just a reset in the top of the CSS


```
img { max-width:100%; }
```
 - max-width overrides the width property
 - If the image size is larger than the container's size → the image gets the entire container's width

13

Flexible Fonts



Flexible Images

Live Demo



Flexible Fonts

- Flexible fonts means proportional font size
 - Based on the context (parent)
 - Instead of pixels use proportional values (em)
- Make all font sizes based on the context
 - To change the font-size of all elements just change the context's font-size

16

Flexible Fonts (2)

- Making fonts "responsive" needs a little math

- `ems` `target / root`
- `1.4375em` `23px / 16px`

```
body { font-size: 16px; }
  body header { font-size: 23px; } // 23 / 16 = 1.4375
```

```
body { font-size: 16px; }
body header { font-size: 1.4375em; }
```

17

Flexible Fonts

Live Demo

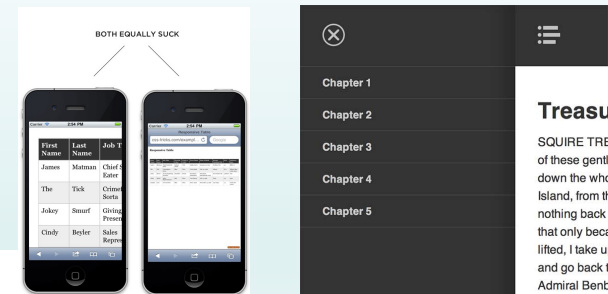


Flexible Tables and Menus

```
/* If the screen size is 601px wide or more, set the font-size of <div> to 80px */
@media screen and (min-width: 601px) {
  div.example {
    font-size: 80px;
  }
}

/* If the screen size is 600px wide or less, set the font-size of <div> to 30px */
@media screen and (max-width: 600px) {
  div.example {
    font-size: 30px;
  }
}
```

18

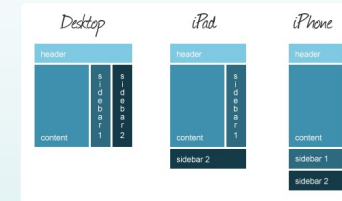


Responsive Tables and Menus

- **Responsive tables**
 - Tables are ugly and not scalable
 - [Demo](#)
- **Responsive menus**
 - Menus take too much space
 - [Demo](#)

21

Media Queries



Responsive Tables

- add a container element with `overflow-x:auto` around the `<table>`

```
<div style="overflow-x:auto;">
  <table>
    ...
  </table>
</div>
```

22

Media Queries

- **Media queries are part of CSS 3**
 - Supported in all major browsers
- **A media query consists of a media type and at least one expression**
 - By using media features like width, height and color
- **Media Queries change the presentation of content**
 - Not the content itself

24

Media Queries

- Media queries apply CSS styles on certain conditions (media type and expression)
- Syntax:

```
@media not|only mediatype and (media feature and|or|not mediafeature)
{
    CSS-Code;
}
```

Media Types	
all	Default
print	Use for printers
Screen	Used for computer, tablets, smart-phones

25

Media Queries

```
.box {width: 250px; height: 250px; display: inline-block}
@media only screen and (max-width: 600px) {
    body {
        background-color: lightblue;
    }
}
@media only screen and (max-width: 1024px) {
    .box { width: 300px; height: 300px; }
}
@media only screen and (max-width: 960px) {
    .box { width: 310px; height: 310px;}
}
@media only screen and (max-width: 480px) {
    .box { display: block; width: 95%; height: 95%; }
}
```

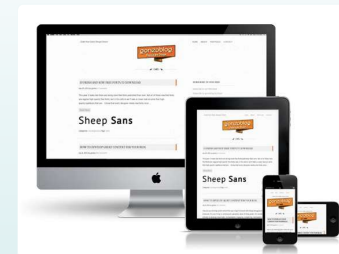
27

Media Queries

Media features	
Height	
Max-height / min-height	The maximum/minimum height of the display area, such as a browser window
Max-width / min-width	The maximum/minimum width of the display area, such as a browser window
Width	
Monochrome	The number of bits per "color" on a monochrome (greyscale) device
Min-resolution	
Max-resolution	

26

Media Queries



Live Demo

BOOTSTRAP

What is Bootstrap?

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

Contents

- Bootstrap Overview – Introduction
- File Structure
- Grid System
- CSS Overview
- Typography
- Tables
- Forms
- Buttons
- Images
- Helper Class

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:

Since Bootstrap 3, the framework consists of Mobile first styles throughout the entire library instead of in separate files.

➤ Browser Support:

It is supported by all popular browsers.



What Bootstrap Package Includes?

➤ Scaffolding: Bootstrap provides a basic structure with Grid System, link styles, background. This is covered in detail in the section Bootstrap Basic Structure

➤ CSS: Bootstrap comes with 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, popovers, and much more. This is covered in detail in the section Layout Components.

Why use Bootstrap?

➤ 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 responsive design in the chapter Bootstrap Responsive Design Provides a clean and uniform solution for building an interface for developers.



What Bootstrap Package Includes?

▪ 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.

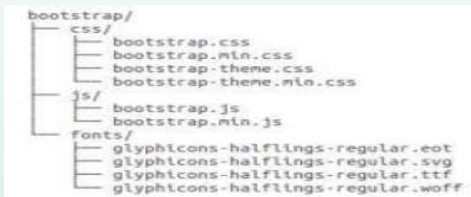
Download Bootstrap

▪ You can download the latest version of Bootstrap from <http://getbootstrap.com/>.

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 is the optional Bootstrap theme.

Basic HTML Template using Bootstrap:

```

<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>
  
```

Basic HTML Template using Bootstrap:

```

<!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
TUTORIALS POINT
Simply Easy Learning 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>
  
```

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 view port 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.

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 columns, and only columns may be immediate children of rows.
- ▶ Predefined grid classes like `.row` and `.col-xs-4` are available for quickly making grid layouts. LESS mixins can also be used for more semantic layouts.
- ▶ Columns create gutters (gaps between column content) via padding. That padding is offset in rows for the first and 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 :

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 :

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 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) { ... }
```

Grid options:

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

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

BASIC GRID STRUCTURE

Following is basic structure of Bootstrap grid:

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

Bootstrap Grid System Example: Stacked-to-horizontal

Let us see a simple grid example with simple layout: two columns

```
<div class="container">
  <div class="row">
    <div class="col-md-6">
      <p>Lorem ipsum dolor sit...</p>
      <p>Sed ut perspiciatis...</p>
    </div>
    <div class="col-md-6">
      <p>Sed ut perspiciatis unde ...</p>
      <p>Neque porro quisquam est...</p>
    </div>
  </div>
</div>
```

BASIC GRID STRUCTURE

Let us see some simple grid examples:

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

OUTPUT:

Lorem ipsum dolor sit amet, consectetur adipiscing 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.
 Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo.

Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium.
 Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem.

Details:

- ▶ `<div class="container">...</div>` element is added to ensure proper centering and maximum width for layout.
- ▶ Once container is added, next you need think in terms of rows. Add `<div class="row">...</div>` and columns `<div class="col-md-6"></div>` inside rows.
- ▶ Every row in the grid is made up of 12 units and you can define the desired size of your columns using those units. In our example we have two columns each made of 6 units wide i.e $6+6=12$.

You can try some more options like `<div class="col-md-3"></div>` and `<div class="col-md-9"></div>` or `<div class="col-md-7"></div>` and `<div class="col-md-5"></div>`

Bootstrap Grid System Example: Medium and Large Device

Here we had used 2 divs and gave them the 50%/50% split at the medium viewport width:

```
<div class="container">
  <div class="row">
    <div class="col-md-6 col-lg-4">
      <p>Lorem ipsum dolor sit ...</p>
      <p>Sed ut perspiciatis unde ...</p>
    </div>
    <div class="col-md-6 col-lg-8">
      <p>Sed ut perspiciatis unde ...</p>
      <p>Neque porro quisquam est ...</p>
    </div>
  </div>
</div>
```

Bootstrap Grid System Example: Mobile, Tablet, Desktops

Now this gives us 3 different column layouts <div class="container">

- On a phone, it will be 25% on the left, and 75% on the right.
- On a tablet, it will be 50%/50% again,
- Large viewport, it will be 33%/66%.
- 3 different layouts for each of the 3 responsive sizes. Check it out in the following example.

OUTPUT:

Lorem ipsum dolor sit amet, consectetur adipiscing 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.
 Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo.

Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo.

Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem.

- But at large design could really be better as a 33%/66%.
- In this case, our 2 divs will go from a 50%/50% split and then up to a 33%/66%. Check it out in the above example.

```
<div class="container">
  <div class="row">
    <div class="col-sm-3 col-md-6 col-lg-8">
      <p>Lorem ipsum dolor sit amet ...</p>
      <p>Sed ut perspiciatis unde ...</p>
    </div>
    <div class="col-sm-9 col-md-6 col-lg-4">
      <p>Sed ut perspiciatis unde ...</p>
      <p>Neque porro quisquam est ...</p>
    </div>
  </div>
</div>
```

OUTPUT:

Lorem ipsum dolor sit amet, consectetur adipiscing 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.
 Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo.

Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo.

Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem.

Responsive images

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

```

```



Typography

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

Headings:

All HTML headings (h1 to h6) are styled in Bootstrap. An example is as 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>
```

Containers:

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

```
<div class="container"> ... </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;
  }
}
```

Typography

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>
```

Typography

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`.

Tables

Basic Table:

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

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

Typography

Code with Example for Lists:

```
<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>
```

OUTPUT:

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

Tables

STRIPED TABLE :

By adding the `.table-striped` class, you will get stripes on rows within the `<tbody>` as seen in the following example:

```
<table class="table table-striped">
  <caption>Striped Table Layout</caption>
  .
  .
  .
</table>
```


Tables

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 class="table table-bordered">
  <caption>Striped Table Layout</caption>
  .
  .
  .
</table>
```

Tables

RESPONSIVE TABLES:

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

```
<div class="table-responsive">
  <table class="table">
    <caption>Responsive Table Layout</caption>
    .
    .
    .
  </table>
</div>
```

Tables

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 class="table table-hover">
  <caption>Striped Table Layout</caption>
  .
  .
  .
</table>
```

Forms

Here we will discuss, 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

Forms

VERTICAL OR BASIC FORM:

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

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

Forms

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

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

Enter Name No file selected. ☐ Check me out

Forms

VERTICAL OR BASIC FORM:

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

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

Name
Enter Name

File input
Browse... No file selected.
Example block-level help text here.

☐ Check me out

Submit

Forms

HORIZONTAL FORM:

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

```
<form class="form-horizontal" role="form">
  <div class="form-group">
    <label for="firstname" class="col-sm-2 control-label">First Name</label>
    <div class="col-sm-10">
      <input type="text" class="form-control"
        id="firstname" placeholder="Enter First Name">
    </div>
  </div>
</form>
```

First Name

Last Name

☐ Remember me

Sign in

Buttons

Here we will discuss about how to use Bootstrap button with examples. Anything that is given a class of .btn will inherit the default look of a gray button with rounded corners.

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.

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:

Buttons

Following example demonstrates all the above button classes:

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

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

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

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

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

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

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

OUTPUT:



Images

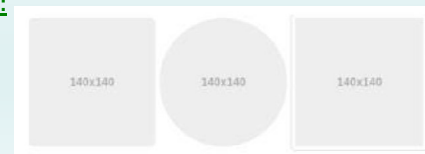
The following example demonstrates this:

```



```

OUTPUT:



Helper Classes

Close icon:

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

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

Close Icon Example ✕

Carets:

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

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

Caret Example ▾

QUESTIONS...

Summary

Responsive design involves

- Fluid layouts
- Flexible Images
- Flexible fonts
- Media queries

Media queries syntax:



```
@media screen and (max-width: 480px) {
  /* Apply styles for small devices (phones) */
}
```