Global settings

Requires HTML5 doctype

Bootstrap makes use of certain HTML elements and CSS properties that require the use of the HTML5 doctype. Include it at the beginning of all your projects.

```
<!DOCTYPE html>
<html lang="en">
  ...
</html>
```

Typography and links

Bootstrap sets basic global display, typography, and link styles. Specifically, we:

- Remove `margin` on the body
- Set `background-color: white;` on the `body`
- Use the `@baseFontFamily`, `@baseFontSize`, and `@baseLineHeight` attributes as our typographic base
- Set the global link color via `@linkColor` and apply link underlines only on `:hover`

These styles can be found within **scaffolding.less**.

Reset via Normalize

With Bootstrap 2, the old reset block has been dropped in favor of Normalize.css, a project by Nicolas Gallagher that also powers the HTML5 Boilerplate. While we use much of Normalize within our **reset.less**, we have removed some elements specifically for Bootstrap.

Default grid system

Live grid example

The default Bootstrap grid system utilizes **12 columns**, making for a 940px wide container without responsive features enabled. With the responsive CSS file added, the grid adapts to be 724px and 1170px wide depending on your viewport. Below 767px viewports, the columns become fluid and stack vertically.

Basic grid HTML

For a simple two column layout, create a `.row` and add the appropriate number of `.span*` columns. As this is a 12-column grid, each `.span*` spans a number of those 12 columns, and should always add up to 12 for each row (or the number of columns in the parent).

```
<div class="row">
  <div class="span4">...</div>
  <div class="span8">...</div>
</div>
```

Given this example, we have `.span4` and `.span8`, making for 12 total columns and a complete row.

Offsetting columns

Move columns to the right using `.offset*` classes. Each class increases the left margin of a column by a whole column. For example, `.offset4` moves `.span4` over four columns.

3 offset 2

3 offset 1

3 offset 2

6 offset 3

```
<div class="row">
  <div class="span4">...</div>
  <div class="span3 offset2">...</div>
</div>
```

Nesting columns

To nest your content with the default grid, add a new `.row` and set of `.span*` columns within an existing `.span*` column. Nested rows should include a set of columns that add up to the number of columns of its parent.

el 1 column
2
2

```
<div class="row">
  <div class="span9">
    Level 1 column
    <div class="row">
      <div class="span6">Level 2</div>
      <div class="span3">Level 2</div>
    </div>
  </div>
</div>
```

Fluid grid system

Live fluid grid example

The fluid grid system uses percents instead of pixels for column widths. It has the same responsive capabilities as our fixed grid system, ensuring proper proportions for key screen resolutions and devices.

1
1
1
1
1
1
1
1
1
1
1
1

4
4
4

4
8

6
6

12

Basic fluid grid HTML

Make any row "fluid" by changing `.row` to `.row-fluid`. The column classes stay the exact same, making it easy to flip between fixed and fluid grids.

```
<div class="row-fluid">
  <div class="span4">...</div>
  <div class="span8">...</div>
</div>
```

Fluid offsetting

Operates the same way as the fixed grid system offsetting: add `.offset*` to any column to offset by that many columns.

4

4 offset 4

3 offset 3
3 offset 3

6 offset 6

```
<div class="row-fluid">
  <div class="span4">...</div>
  <div class="span4 offset2">...</div>
</div>
```

Fluid nesting

Nesting with fluid grids is a bit different: the number of nested columns should not match the parent's number of columns. Instead, each level of nested columns are reset because each row takes up 100% of the parent column.

Fluid 12
Fluid 6
Fluid 6

```
<div class="row-fluid">
  <div class="span12">
    Fluid 12
    <div class="row-fluid">
      <div class="span6">Fluid 6</div>
      <div class="span6">Fluid 6</div>
    </div>
  </div>
</div>
```

Layouts

Fixed layout

Provides a common fixed-width (and optionally responsive) layout with only `<div class="container">` required.


```
<body>
  <div class="container">
    ...
  </div>
</body>
```

Fluid layout

Create a fluid, two-column page with `<div class="container-fluid">`—great for applications and docs.

```
<div class="container-fluid">
  <div class="row-fluid">
    <div class="span2">
      <!--Sidebar content-->
    </div>
    <div class="span10">
      <!--Body content-->
    </div>
  </div>
</div>
```

Responsive design

Enabling responsive features

Turn on responsive CSS in your project by including the proper meta tag and additional stylesheet within the `<head>` of your document. If you've compiled Bootstrap from the Customize page, you need only include the meta tag.

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link href="assets/css/bootstrap-responsive.css" rel="stylesheet">
```

Heads up! Bootstrap doesn't include responsive features by default at this time as not everything needs to be responsive. Instead of encouraging developers to remove this feature, we figure it best to enable it as needed.

About responsive Bootstrap

Media queries allow for custom CSS based on a number of conditions—ratios, widths, display type, etc—but usually focuses around `min-width` and `max-width`.

- Modify the width of column in our grid
- Stack elements instead of float wherever necessary
- Resize headings and text to be more appropriate for devices

Use media queries responsibly and only as a start to your mobile audiences. For larger projects, do consider dedicated code bases and not layers of media queries.

Supported devices

Bootstrap supports a handful of media queries in a single file to help make your projects more appropriate on different devices and screen resolutions. Here's what's included:

Label	Layout width	Column width	Gutter width
Large display	1200px and up	70px	30px
Default	980px and up	60px	20px
Portrait tablets	768px and above	42px	20px
Phones to tablets	767px and below	Fluid columns, no fixed widths	
Phones	480px and below	Fluid columns, no fixed widths	

```
/* Large desktop */
@media (min-width: 1200px) { ... }

/* Portrait tablet to landscape and desktop */
@media (min-width: 768px) and (max-width: 979px) { ... }

/* Landscape phone to portrait tablet */
@media (max-width: 767px) { ... }

/* Landscape phones and down */
@media (max-width: 480px) { ... }
```

Responsive utility classes

For faster mobile-friendly development, use these utility classes for showing and hiding content by device. Below is a table of the available classes and their effect on a given media query layout (labeled by device). They can be found in `responsive.less`.

Class	Phones 767px and below	Tablets 979px to 768px	Desktops Default
`.visible-phone`	Visible	Hidden	Hidden

Class	Phones 767px and below	Tablets 979px to 768px	Desktops Default
`.visible-tablet`	Hidden	Visible	Hidden
`.visible-desktop`	Hidden	Hidden	Visible
`.hidden-phone`	Hidden	Visible	Visible
`.hidden-tablet`	Visible	Hidden	Visible
`.hidden-desktop`	Visible	Visible	Hidden

When to use

Use on a limited basis and avoid creating entirely different versions of the same site. Instead, use them to complement each device's presentation. Responsive utilities should not be used with tables, and as such are not supported.

Responsive utilities test case

Resize your browser or load on different devices to test the above classes.

Visible on...

Green checkmarks indicate that class is visible in your current viewport.

Phone
Tablet
Desktop✔ Desktop

Hidden on...

Here, green checkmarks indicate that class is hidden in your current viewport.

Phone✔ Phone
Tablet✔ Tablet
Desktop

Back to top