
Bootstrap

Dave Winer and Jake Spurlock

Bootstrap

by Dave Winer and Jake Spurlock

Copyright © 2010 Jake Spurlock. All rights reserved.

Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (<http://my.safaribooksonline.com>). For more information, contact our corporate/institutional sales department: 800-998-9938 or corporate@oreilly.com.

Editors: Simon St. Laurent and Meghan Blanchette

Indexer:

Production Editor: Marisa LaFleur

Cover Designer: Karen Montgomery

Copyeditor:

Interior Designer: David Futato

Proofreader: FIX ME!

Illustrator: Robert Romano

May 2013: First Edition

Revision History for the First Edition:

YYYY-MM-DD: First release

See <http://oreilly.com/catalog/errata.csp?isbn=9781449343910> for release details.

Nutshell Handbook, the Nutshell Handbook logo, and the O'Reilly logo are registered trademarks of O'Reilly Media, Inc. !!FILL THIS IN!! and related trade dress are trademarks of O'Reilly Media, Inc.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and O'Reilly Media, Inc., was aware of a trademark claim, the designations have been printed in caps or initial caps.

While every precaution has been taken in the preparation of this book, the publisher and authors assume no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein.

ISBN: 978-1-449-34391-0

[?]

Table of Contents

Foreword.....	vii
Preface.....	ix
1. Bootstrap Scaffolding.....	1
What is Bootstrap	1
Bootstrap File Structure	2
Basic HTML Template	2
Global Styles	3
Default Grid System	3
Basic grid HTML	4
Offsetting Columns	4
Nesting columns	5
Exercercise 1	5
Fluid Grid System	6
Container Layouts	7
Responsive Design	7
What is Responsive Design?	8
2. Bootstrap CSS.....	11
Typography	11
Headings	11
Emphasis classes	13
Lists	17
Code	20
Tables	20
Optional Table Classes	21
Table Row Classes	23
Forms	23
Optional Form Layouts	24

Supported Form Controls	26
Extending Form Controls	29
Form Control Sizing	31
Form Control States	35
Buttons	36
Button Sizes	38
Disabled Button Styling	39
Images	40
Icons	40
Glyphicon Attribution	41
Usage	42
3. Bootstrap Layout Components.....	45
Dropdown Menus	45
Options	46
Button Groups	46
Button Groups as Radio and Checkboxes	48
Buttons With Dropdowns	48
Split Button Dropdowns	49
Dropup Menus	50
Navigation Elements	51
Tabular Navigation	51
Basic Pills Navigation	51
Stackable Navs	52
Dropdowns	53
Navigation Lists	54
Tabbable Navigation	55
Navbar	58
Nav Links	59
Forms	60
Navbar Menu Variations	60
Breadcrumbs	63
Pagination	64
Pager	66
Labels and Badges	67
Badges	67
Typographic Elements	68
Page Header	68
Thumbnails	69
Alerts	70
Progress bars	72
Media Object	73

Misc	76
Helper Classes	77
4. Bootstrap Javascript Plugins.....	79
Overview	79
Programmatic API	80
Transitions	80
Modal	81
Using Modal through Data Attributes	82
Modal Options	82
Methods	83
Events	83
Dropdown	84
Dropdown Usage via the Data API	84
Dropdown Usage via Javascript	85
Methods	85
Scrollspy	85
Usage	85
Scrollspy Methods	86
Options	86
Events	86
Toggleable Tabs	86
Usage	87
Events	88
Tooltips	88
Usage	88
Options	89
Methods	89
Show	89
Popover	90
Usage	90
Options	90
Methods	91
Alerts	91
Usage	92
Methods	92
Events	92
Buttons	92
Loading State	93
Single Toggle	93
Checkbox Buttons	93
Radio Buttons	94

Usage	94
Methods	94
Collapse	95
Usage	96
Options	96
Methods	97
Events	97
Carousel	97
Usage	98
Options	98
Methods	99
Events	99
Typeahead	100
Usage	100
Options	101
Affix	101
Usage	101
Options	102
5. Using Bootstrap.....	103
Github Project	103
Customizing Bootstrap	103
Using LESS	104
Text Snippets	106
Photoshop Templates	107
Themes	108
Built with Bootstrap	108
Conclusion	108
Index.....	111

Foreword

I was a software developer before there was a Mac, so I remember something not a lot of programmers do. I remember how the tech industry reacted to it. And for the most part, it was with a fair amount of skepticism. And the interesting thing is that the negative things people say about Bootstrap today sound exactly like the negative things people said about the Mac in 1984. And in both cases, the things that people didn't like were what made them important.

What the Mac realized is that there are a set of things that all software has to do, so why shouldn't they all do them the same way? If they did, software would be easier to develop and debug, but more important — it would be easier to use. If there was only one way to do menus, then once a user learned how to use the menus of one app, they'd already know how to use the menus of all others. Same with scrollbars, windows, the keyboard, the mouse, printing, sound.

The reason programmers didn't like it, and I was one of them, was that they took what we did and commoditized it. Further, there were limits to the one-size-fits-all approach. There were some apps that didn't take to the UI standards very well. What to do about them? Well, you adapted, that's what you did.

This is a well-known technical process called factoring. If you see yourself doing something over and over, do it one more time, really well, and work on the API so it's really easy and flexible, and that's it. You never do it again. It's how you build ever-taller buildings out of software. And what was the leading-edge five years ago, is baked into the operating system today. Progress. It's a wonderful thing! :-)

The same patterns are observable in the web. In fact, it's kind of sad how much of a repeat it is, how backward today's development environment is compared to the one envisioned by the Mac. But at least Bootstrap is out there doing the factoring. If I want to put up a menu, I can just use their code that does menus. Sure, my menu looks like all the others, but that's a good thing, for users. No need to learn a second or third way to use a menu.

That this is needed, desperately needed, is indicated by the incredible uptake of Bootstrap. I use it in all the server software I'm working on. And it shows through in the templating language I'm developing, so everyone who uses it will find it's "just there" and works, any time you want to do a Bootstrap technique. Nothing to do, no libraries to include. It's as if it were part of the hardware. Same approach that Apple took with the Mac OS in 1984.

Like all important technologies, Bootstrap is "good enough" but not too good. In other words, the designers, Mark Otto and Jacob Thornton, could have factored more than they did, and created something more compact and perhaps more elegant, that wasn't nearly as approachable. The great thing about Bootstrap for a guy like me, who has been busy building software behind websites, is that it solves a whole bunch of problems that we all have when putting a user interface on those sites.

However I think that will turn out to be just the beginning. I see the opportunity for Bootstrap to become an integral part of the web. A toolkit that you can count on being present in every environment you work in. Further, someday, perhaps soon, designers will be able to plug in "skins" for Bootstrap, that transform the appearance of a site without any modification to the code or to its styles or scripts. I don't see any limits to what can be done with Bootstrap. Rather than being a replacement for designers, it creates opportunities for designers to have more power and reach.

These days, part of the maturing process of any new technology is the release of its O'Reilly book. Now Bootstrap has one. The author, Jake Spurlock, a web developer, has built a bunch of sites with Bootstrap, has spoken at conferences about it, and he credits me for getting him started with a series of enthusiastic blog posts I wrote, as I was discovering its power and elegance. Now, I can happily turn you over to Jake's able hands, to show you how the Bootstrap magic works, so you too can help move the web development world forward.

—Dave Winer, editor, Scripting News, January 2013, New York, NY

Preface

Bootstrap is a front-end framework for building responsive websites. Whether it is application frameworks, blogs, or other CMS applications, Bootstrap can be a good fit as it can be as vanilla as you like. It's combination of HTML, CSS, and Javascript make it easy to build robust sites, without adding a lot of code. With a default grid system, layouts come together with ease, and the styling of buttons, navs, and tables make basic markup look great from the get go. A dozen or so Javascript plugins catapult you into adding interactive elements to your site.

Who This Book is For

This books is mostly for people that have a good handle on HTML/CSS and Javascript, and are curious to get started with building responsive sites, adding the Bootstrap Javascript plugins, or building sites faster by using this popular open-source framework.

Who This Book is Not For

This book is not for people that get all they need out of the Bootstrap online documentation. Like a lot of people, the online docs are where I got started, building my first site with Bootstrap 1.3, and then upgrading it to Bootstrap 1.4. After that, a big project was built with Bootstrap 2.0 and on. If you are comfortable writing semantic HTML, then jumping into Bootstrap should be easy for you to get started with.

What This Book Will Do For You

With some background in writing HTML/CSS, and Javascript, this book will help you get off the ground writing some flexible code for responsive websites. In practical terms, the concepts and code syntax should come easily as it follows the patterns for writing semantic HTML and CSS anyways.

How This Book Works

This book tries to built a site with Bootstrap, starting at the foundation of the project, the file structure, moving up through the grid system, and layout types into HTML elements and styling like forms, tables, and buttons. Once the walls are up, we move into the aesthetic elements like navbars, breadcrumbs, and media objects. After that, into the Javascript elements that provide the interaction for a site like dropdowns, the carousel, and modals.

Why I Wrote This Book

I'm not an Bootstrap expert hoping to create more Bootstrap experts to get a lot of work done.

I'm a developer and writer who encountered Bootstrap through a post on Dave Winer's blog, and I thought that it would be cool to apply it to a new site that I was working on and since felt compelled to share some of that. I'm hoping that the path I followed will work for other people, probably with variations, and that a book written from a beginner's perspective (and vetted by experts) would help more people find and enjoy Bootstrap.

Other Resources

This book may not be the best way for you to learn Bootstrap. It all depends on what you want to learn and why.

If your primary interest is to get started building Bootstrap websites, the online documentation at <http://twitter.github.com/bootstrap/> may likely suit you perfectly. The authors, Jacob Thornton and Mark Otto have been meticulous in providing examples of the codebase, and HTML code samples, and more to kick start your project. It is top notch, and where I have gathered the structure for this book.

If you want to contribute to the work of the open source project, you can submit pull requests, or use the issue tracker on the Github project, <http://github.com/twitter/bootstrap/>, for updates, downloads, documentation, and more.

Are You Sure You Want Bootstrap?

If you are simply looking for Javascript plugins, or a CSS reset, Bootstrap maybe overkill. If you aren't in love with some of the default interface elements that can be overwritten easily, or just strip out the associated tags. If you are looking for an easy way to build fast, responsive websites, Bootstrap is a great way to get going. I use it on all of my projects, and I'm really happy with it.

Conventions Used in This Book

The following typographical conventions are used in this book:

Italic

Indicates new terms, URLs, email addresses, filenames, and file extensions.

Constant width

Used for program listings, as well as within paragraphs to refer to program elements such as variable or function names, statements, and keywords.

Constant width bold

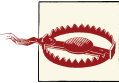
Shows commands or other text that should be typed literally by the user.

Constant width italic

Shows text that should be replaced with user-supplied values or by values determined by context.



This icon signifies a tip, suggestion, or general note.



This icon indicates a warning or caution.

Using Code Examples

The examples in this book are meant to teach basic concepts in small bites. While you may certainly borrow code and reuse it as you see fit, you won't be able to take the code of this book and build a stupendous application instantly (unless perhaps you have an unusual fondness for bacon and cats). You should, however, be able to figure out the steps you need to take to build a great website.

You can download the code from the Examples link on the book's page at <http://shop.oreilly.com/product/0636920027867.do>.

This book is here to help you get your job done. In general, if this book includes code examples, you may use the code in this book in your programs and documentation. You do not need to contact us for permission unless you're reproducing a significant portion of the code. For example, writing a program that uses several chunks of code from this book does not require permission. Selling or distributing a CD-ROM of examples from O'Reilly books does require permission. Answering a question by citing this book and quoting example code does not require permission. Incorporating a significant amount

of example code from this book into your product's documentation does require permission.

We appreciate, but do not require, attribution. An attribution usually includes the title, author, publisher, and ISBN. For example: "*Bootstrap, Responsive Web Development*, by Jake Spurlock (O'Reilly). Copyright 2013 Jake Spurlock, 9781449343910."

If you feel your use of code examples falls outside fair use or the permission given above, feel free to contact us at permissions@oreilly.com.

Help This Book Grow

While I hope that you will enjoy reading this book and learn from it, I also hope that you can contribute to helping other readers learn Bootstrap here. You can help your fellow readers in a number of ways:

- If you find specific technical problems, bad explanations, or things that can be improved, please report them through the errata system at <http://oreilly.com/catalog/errata.csp?isbn=0636920027867>.
- If you like (or don't like) the book, please leave reviews. The most visible places to do so are on Amazon.com (or its international sites) and at the O'Reilly page for the book at <http://shop.oreilly.com/product/0636920027867.do>. Detailed explanations of what worked and what didn't work for you (and the broader target audience of programmers new to Bootstrap) are helpful to other readers and to me.
- If you find you have much more you want to say about Bootstrap, please consider sharing it, whether on the Web, in a book of your own, in training classes, or in whatever form you find easiest.

I'll update the book for errata, and try to address issues raised in reviews. Even once the book is *complete*, I may still add some extra pieces to it. If you purchased it as an ebook, you'll receive these updates for free at least up to the point where it's time for a whole new edition. I don't expect that new edition declaration to come quickly, however, unless the Bootstrap world changes substantially.

Hopefully this book will engage you enough to make you consider sharing.

Safari® Books Online



Safari Books Online is an on-demand digital library that delivers expert **content** in both book and video form from the world's leading authors in technology and business.

Technology professionals, software developers, web designers, and business and creative professionals use Safari Books Online as their primary resource for research, problem solving, learning, and certification training.

Safari Books Online offers a range of **product mixes** and pricing programs for **organizations**, **government agencies**, and **individuals**. Subscribers have access to thousands of books, training videos, and prepublication manuscripts in one fully searchable database from publishers like O'Reilly Media, Prentice Hall Professional, Addison-Wesley Professional, Microsoft Press, Sams, Que, Peachpit Press, Focal Press, Cisco Press, John Wiley & Sons, Syngress, Morgan Kaufmann, IBM Redbooks, Packt, Adobe Press, FT Press, Apress, Manning, New Riders, McGraw-Hill, Jones & Bartlett, Course Technology, and dozens **more**. For more information about Safari Books Online, please visit us **online**.

How to Contact Us

Please address comments and questions concerning this book to the publisher:

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (in the United States or Canada)
707-829-0515 (international or local)
707-829-0104 (fax)

We have a web page for this book, where we list errata, examples, and any additional information. You can access this page at <http://www.oreilly.com/catalog/<catalog page>>.

To comment or ask technical questions about this book, send email to bookquestions@oreilly.com.

For more information about our books, courses, conferences, and news, see our website at <http://www.oreilly.com>.

Find us on Facebook: <http://facebook.com/oreilly>

Follow us on Twitter: <http://twitter.com/oreillymedia>

Watch us on YouTube: <http://www.youtube.com/oreillymedia>

Acknowledgments

Many thanks to Dave Winer for interesting me in Bootstrap in the first place, and to Simon St. Laurent for the opportunity to write this. Detailed feedback from my friends Roseanne Fallin, and Tony Quartorolo has made it possible, I hope that this book can

get readers started on the right track. Thanks to Melissa Morgan for letting my take a few risks and develop the way that I like at MAKE too.

In particular, thanks to my wonderful wife Melissa for putting up with me, and encouraging me to finish. To my son Rush for understanding that I needed to “Work”, and to my daughter Hailey for the warm smiles and huge hugs.

Bootstrap Scaffolding

What is Bootstrap

Bootstrap is an Open Source product from Mark Otto and Jacob Thornton who, when initially released were both employees at Twitter. There was a need to standardize the front end toolsets of engineers across the company. In the launch blog post, Mark Otto introduces the project like this:

In the earlier days of Twitter, engineers used almost any library they were familiar with to meet front-end requirements. Inconsistencies among the individual applications made it difficult to scale and maintain them. Bootstrap began as an answer to these challenges and quickly accelerated during Twitter's first Hackweek. By the end of Hackweek, we had reached a stable version that engineers could use across the company.

— Mark Otto
*[https://dev.twitter.com/blog/
bootstrap-twitter](https://dev.twitter.com/blog/bootstrap-twitter)*

Since Bootstrap launched in August, 2011 it has taken off in popularity. It has evolved away from being an entirely CSS driven project to include a host of Javascript plugins, and icons that go hand in hand with forms and buttons. At its base, it allows for responsive web design, and features a robust 12 column, 940px wide grid. One of the highlights is the build tool on [Bootstrap's](<http://getbootstrap.com>) website where you can customize the build to suit your needs, choosing what CSS and Javascript features that you want to include on your site. All of this, allows front-end web development to be catapulted forward, building on a stable foundation of forward-looking design, and development. Getting started with Bootstrap is as simple as dropping some CSS and Javascript into the root of your site.

Starting a project new, Bootstrap comes with a handful of useful elements to get you started. Normally, when I start a project, I start with tools like [Eric Meyer's CSS reset](#), and get going on my web project. With Bootstrap, you just need to include the

bootstrap.css CSS file, and optionally the bootstrap.js Javascript file into your website and you are ready to go.

Bootstrap File Structure

```
bootstrap/
├── css/
│   ├── bootstrap.css
│   └── bootstrap.min.css
├── js/
│   ├── bootstrap.js
│   └── bootstrap.min.js
├── img/
│   ├── glyphs-glyphicons-halflings.png
│   └── glyphs-glyphicons-halflings-white.png
└── README.md
```

The Bootstrap download includes three folders: css, js, and img. For simplicity, add these to the root of your project. Included are also minified versions of the CSS and Javascript. Both the uncompressed and the minified versions do not need to be included. For the sake of brevity, I use the uncompressed during development, and then switch to the compressed version in production.

Basic HTML Template

Normally, a web project looks something like this:

Basic HTML Layout.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Bootstrap 101 Template</title>
  </head>
  <body>
    <h1>Hello, world!</h1>
  </body>
</html>
```

With Bootstrap, we simply include the link to the CSS stylesheet, and the Javascript.

Basic Bootstrap Template.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Bootstrap 101 Template</title>
    <link href="css/bootstrap.min.css" rel="stylesheet">
  </head>
  <body>
    <h1>Hello, world!</h1>
```



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



NOTE Don't forget the HTML 5 Doctype.

By including `<!DOCTYPE html>`, all modern browsers are put into standards mode.

Global Styles

With Bootstrap, a number of items come prebuilt. Instead of using the old reset block that was part of the Bootstrap 1.0 tree, Bootstrap 2.0 uses [Normalize.css](#), a project from [Nicolas Gallagher](#) that is part of the [HTML5 Boilerplate](#). This is included in the Bootstrap.css file.

In particular, these default styles give special treatment to typography and links.

- `margin` has been removed from the body, and content will snug up to the edges of the browser window.
- `background-color: white;` is applied to the body
- Bootstrap is using the `@baseFontFamily`, `@baseFontSize`, and `@baseLineHeight` attributes as our typographic base. This allows the height of headings, and other content around the site to maintain a similar line height.
- Bootstrap sets the global link color via `@linkColor` and applies link underlines only on `:hover`



Remember, if you don't like the colors, or want to change a default, this can be done in you can change globals in any of the .less files. For this, update the `scaffolding.less` file, or overwrite colors in your own stylesheet that have on the site.

Default Grid System

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, for example, on tablets and smaller devices the columns become fluid and stack vertically. At the default width, each column is 60 pixels wide, offset 20 pixels to the left.

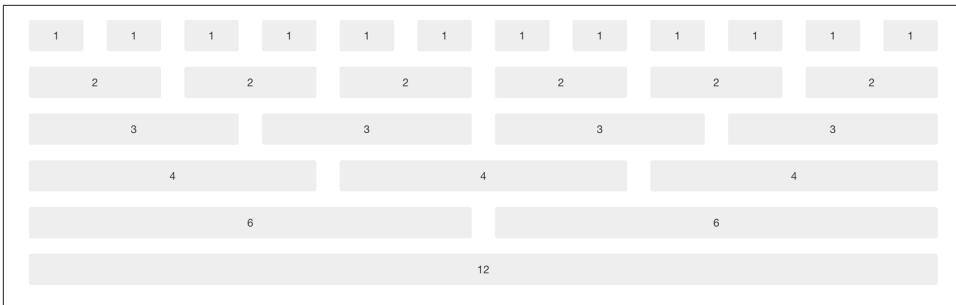


Figure 1-1. Default Grid

Basic grid HTML

To create a simple layout, create a container with a `div` that has a class of `.row`, and add the appropriate amount of `.span*` columns. Since we have 12-column grid, we just need to have the amount of `.span*` columns add up to twelve. We could use a 3-6-3 layout, 4-8, 3-5-4, 2-8-2, we could go on and on, but I think you get the gist.

Basic Grid Layout.

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

In the above example, we have `.span8`, and a `.span4` adding up to 12.

<http://jsfiddle.net/whyisjake/b3X7n/embedded/result/>

Offsetting Columns

You can move columns to the right using the `.offset*` class. Each class moves the span over that width. So an `.offset2` would move a `.span7` over two columns.

Offsetting Columns.

```
<div class="row">
  <div class="span2">...</div>
  <div class="span7 offset2">...</div>
</div>
```

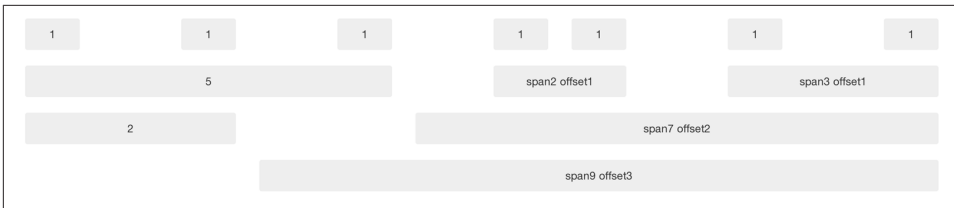


Figure 1-2. Offset Grid

Nesting columns

To nest your content with the default grid, inside of a `.span*`, simply add a new `.row` with enough `.span*` that add up the number of spans of the parent container.

So, let's say that you have a two columns layout, with a `span8`, and a `span4`, and you want to embed a two column layout inside of the layout, what spans would you use? For a four column layout?

Exercise 1

Create a table that looks like this:

Table 1-1. ORM Employees

First	Last
Sanders	Kleinfeld
Karen	Tripp
Adam	Zaremba

Write your solution here:

```

First Last
Sanders Kleinfeld
Karen Tripp
Adam Zaremba

```

Solution

Your markup should look something like this:

```

.ORM Employees
[options="header"]
|=====
|First  | Last
|Sanders| Kleinfeld
|Karen  | Tripp

```

```
| Adam | Zaremba  
|=====
```

Nesting Columns.

```
<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>
```

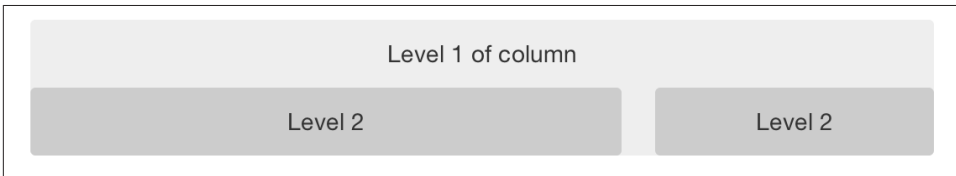


Figure 1-3. Nesting Grid

<http://jsfiddle.net/whyisjake/EEWQ2/embedded/result/>

Fluid Grid System

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. You can 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. To offset, you operate in the same way as the fixed grid system works by adding `.offset*` to any column to shift by your desired number of columns.

<http://jsfiddle.net/whyisjake/327R3/embedded/result/>

Fluid Column Layout.

```
<div class="row-fluid">  
  <div class="span4">...</div>  
  <div class="span8">...</div>  
</div>  
  
<div class="row-fluid">  
  <div class="span4">...</div>  
  <div class="span4 offset2">...</div>  
</div>
```

Nesting a fluid grid is a little different. Since we are using percentages, each `.row` resets the column count to 12. For example, If you were inside a `.span8`, instead of two `.span4` elements to divide the content in half, you would use two `.span6` divs. This is the case for responsive content, as we want the content to fill 100% of the container.

Nesting Fluid Column Layout.

```
<div class="row-fluid">
  <div class="span8">
    <div class="row">
      <div class="span6">...</div>
      <div class="span6">...</div>
    </div>
  </div>
</div>
```

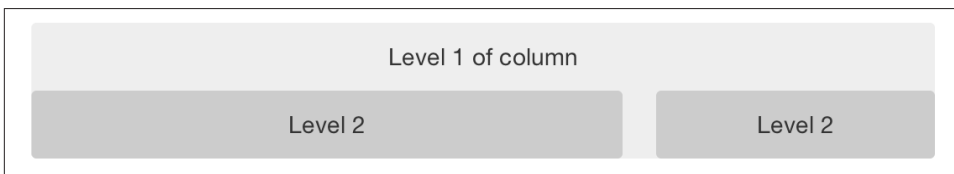


Figure 1-4. Nesting Fluid Grid

Container Layouts

To add a fixed width, centered layout to your page, simply wrap the content in `<div class="container">...</div>`. If you would like to use a fluid layout, but want to wrap everything in a container, use the following: `<div class="container-fluid">...</div>`. Using a fluid layout is great when you are building applications, administration screens and other related projects.

Responsive Design

To turn on the responsive features of Bootstrap, you need to add a meta tag to the `<head>` of your webpage. If you haven't downloaded the compiled source, you will also need to add the responsive CSS file. An example of required files looks like this:

Responsive Meta Tag and CSS.

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

```
<link href="/css/bootstrap-responsive.css" rel="stylesheet">
</head>
```



Heads up! If you get started and are finding that the Bootstrap responsive features aren't working, make sure that you have these tags. The responsive features aren't added by default at this time because not everything needs to be responsive. Instead of encouraging developers to remove this feature, the authors of Bootstrap opted that it was best to enable it as needed.

What is Responsive Design?

Responsive design is a method for taking all of the existing content that is on the page, and optimizing it for the device that is viewing it. For example, the desktop not only gets the normal version of the website, but might get also get a widescreen layout, optimized for the larger displays that many people have attached to their computers. Tablets get an optimized layout, taking advantage of the portrait or landscape layouts of those devices. And then with phones, you can target the much narrower width of phones. To target these different widths, Bootstrap uses CSS media queries to measure the width of the browser viewport, and then using conditionals, change which parts of the stylesheets are loaded. Using the width of the browser viewport, Bootstrap can then optimize the content using a combination of ratios, widths, but mostly falls on *min-width* and *max-width* properties..

At the core, Bootstrap supports five different layouts, each relying on CSS media queries. The largest layout has columns that are 70 pixels wide, contrasting the 60 pixels of the normal layout. The tablet layout brings the columns to 42 pixels wide, and when narrower than that, each column goes fluid, meaning the columns are stacked vertically and each column is the full width of the device.

Table 1-2. Responsive Media Queries

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	

To add custom CSS based on the media query, you can either include all rules in one CSS file, via the media queries below, or use entirely different CSS files.

CSS media queries in the CSS stylesheet.

```

/* 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) { ... }

```

For a larger site, you might want to separate them into separate files. In the HTML file, you can call them with the link tag in the head of your document. This is useful for keeping file sizes smaller, but does potentially increase the HTTP requests if being responsive.

CSS media queries via the link tag in the HTML <head>.

```

<link rel="stylesheet" href="base.css" />
<link rel="stylesheet" media="(min-width:1200px)" href="large.css" />
<link rel="stylesheet" media="(min-width:768px) and (max-width: 979px)" href="tablet.css" />
<link rel="stylesheet" media="(max-width: 767px)" href="tablet.css" />
<link rel="stylesheet" media="(max-width: 480px)" href="phone.css" />

```

Helper Classes

Bootstrap also includes a handful of helper classes for doing responsive development. It would be best practice to use these sparingly. A couple of use cases that I have seen involve loading custom elements based on certain layouts. Perhaps you have a really nice header on the main layout, but on mobile you want to pare it down, leaving only a few of the elements. In this scenario, you could use the `.hidden-phone` class to hide either parts, or entire dom elements from the the header.

Table 1-3. Media Queries Helper Classes

Class	Phones	Tablets	Desktops
<code>.visible-phone</code>	Visible	Hidden	Hidden
<code>.visible-tablet</code>	Hidden	Visible	Hidden
<code>.visible-desktop</code>	Hidden	Hidden	Visible
<code>.hidden-phone</code>	Hidden	Visible	Visible
<code>.hidden-tablet</code>	Visible	Hidden	Visible
<code>.hidden-desktop</code>	Visible	Visible	Hidden

Regarding mobile development, there are two major ways that you could look at doing development. The mantra that a lot of people are shouting now, is that you should start with mobile, build to that platform, and let the desktop follow. Bootstrap almost forces the opposite, where you would create a full featured desktop site that “just works”.

If you are looking for a strictly mobile framework, Bootstrap is still a great resource.

At the core of Bootstrap are a set of basic HTML elements that have been styled in such a way that allow for easy enhancement via classes and user styles.

Typography

Starting with Typography, with the default font stack, Bootstrap uses Helvetica Neue, Helvetica, Arial, and sans-serif. These are all standard fonts, and included as defaults on all major computers, falling back to `sans-serif`, the catch all to tell the browser to use the default font that the user has decided. All body copy has the `font-size` set at 14 pixels, with the `line-height` set at 20 pixels. The `<p>` tag has a `margin-bottom` of 10 pixels, or half the `line-height`.

Headings

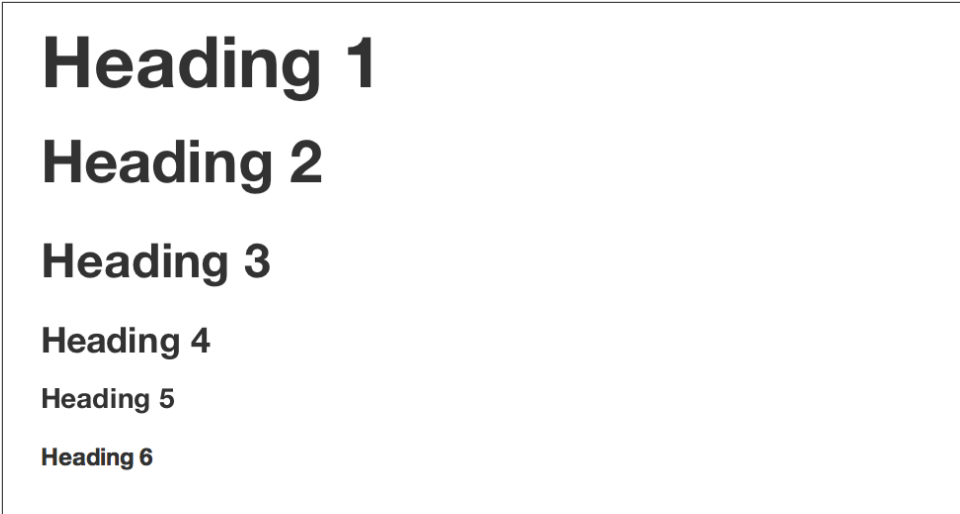


Figure 2-1. Headings

All six standard heading levels have been styled in Bootstrap, with the `<h1>` at 36 pixels high, and the `<h6>` down to 12 pixels in height (for reference, body text is 14 pixels in height by default). In addition, to add an inline sub-heading to any of the headings, simply add `<small>` around any of the elements, and you will get smaller text, in a lighter color. In the case of the `<h1>`, the small text is 24 pixels tall, normal font weight (i.e., not bold), and gray instead of black.

```
h1 small {  
  font-size:24px;  
  font-weight:normal;  
  line-height:1;  
  color:#999;  
}
```

Lead body copy

To add some emphasis to a paragraph, add `class="lead"`. This will give you larger font size, lighter weight, and a taller line height. This is usually used for the first few paragraphs in a section, but can really be used anywhere.

```
<p class="lead">Bacon ipsum dolor sit amet tri-tip pork loin ball tip frankfurter swine boudin mea
```

Lead Example

Bacon ipsum dolor sit amet tri-tip pork loin ball tip frankfurter swine boudin meatloaf shoulder short ribs cow drumstick beef jowl. Meatball chicken sausage tail, kielbasa strip steak turducken venison prosciutto. Chuck filet mignon tri-tip ribeye, flank brisket leberkas. Swine turducken turkey shank, hamburger beef ribs bresaola pastrami venison rump.

Figure 2-2. Lead Body Copy Classes

Emphasis

In addition to using the `<small>` tag within headings, as discussed above, you can also use it with body copy. When `<small>` is applied to body text, the font shrinks to 85% of its original size.

Bold

To add emphasis to text, simply wrap it in a `` tag. This will add `font-weight: bold` to the selected text.

Italics

For italics, wrap your content in the `` tag. “em” derives from the word “emphasis”, and is meant to add stress to your text.



You might be thinking to yourself, why not just use the `` or `<i>` tags instead of `` or ``. In HTML5, `` is meant to highlight words or phrases without conveying additional importance—for example, key terms or names—while `<i>` is mostly for voice, technical terms, internal dialogue, etc. For more information about the semantic changes to `` and `<i>`, check out [W3.org's article](#).

Emphasis classes

Along with `` and ``, Bootstrap offers a few other classes that can be used to provide emphasis. These could be applied to paragraphs, or spans.

Emphasis Classes.

```
<p class="muted">This content is muted</p>
<p class="text-warning">This content carries a warning class</p>
<p class="text-error">This content carries an error class</p>
<p class="text-info">This content carries an info class</p>
```

```
<p class="text-success">This content carries a success class</p>
<p>This content has <em>emphasis</em>, and can be <strong>bold</strong></p>
```

Bootstrap Emphasis Classes

This content is muted

This content carries a warning class

This content carries an error class

This content carries an info class

This content carries a success class

This content has *emphasis*, and can be **bold**

Figure 2-3. Emphasis Classes

Abbreviations

The HTML `<abbr>` element provides markup for abbreviations or acronyms, like WWW or HTTP. By marking up abbreviations, you can give useful information to browsers, spell checkers, translation systems or search engines. Bootstrap styles `<abbr>` elements with a light dotted border along the bottom, and reveals the full text on hover (as long as you add that text to the `<abbr>` title attribute).

Abbreviation Example.

```
<abbr title="Real Simple Syndication">RSS</abbr>
```

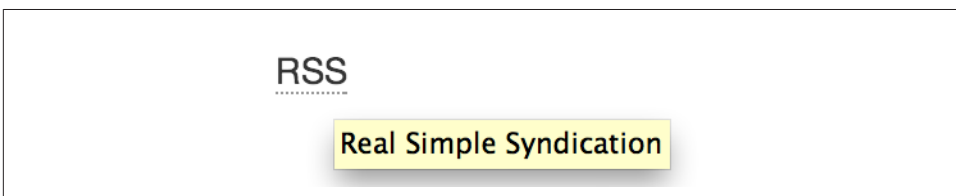


Figure 2-4. Emphasis Classes

Add `.initialism` to an `<abbr>` for a slightly smaller font size.

Abbreviation Example.

```
<abbr title="Real Simple Syndication">RSS</abbr>
```

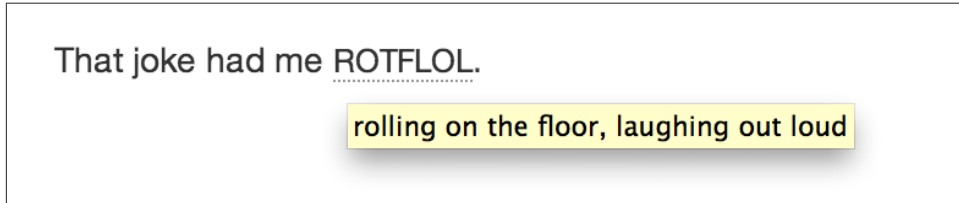


Figure 2-5. *Emphasis Classes*

Addresses

Adding `<address>` elements to your page can help screen readers and search engines locate any physical addresses and phone numbers in the text. It can also be used to mark up email addresses. Since the `<address>` defaults to `display: block`; you'll need to use `
` tags to add line breaks to the enclosed address text (for example, to split the street address and city onto separate lines).

Address Markup.

```
<address>  
  <strong>O'Reilly Media, Inc.</strong><br>  
  1005 Gravenstein HWY North<br>  
  Sebastopol, CA 95472<br>  
  <abbr title="Phone">P:</abbr> <a href="tel:+17078277000">(707) 827-7000</a>  
</address>
```

```
<address>  
  <strong>Jake Spurlock</strong><br />  
  <a href="mailto:#">flast@oreilly.com</a>  
</address>
```

O'Reilly Media, Inc.
1005 Gravenstein HWY North
Sebastopol, CA 95472
P: (707) 827-7000

Jake Spurlock
flast@oreilly.com

Figure 2-6. Address Tag

Blockquotes

To add blocks of quoted text to your document—or for any quotation that you want to set apart from the main text flow—add the `<blockquote>` tag around the text. For best results, and for line breaks, wrap each subsection in a `<p>` tag. Bootstrap's default styling indents the text, and adds a thick grey border along the left side. To identify the source of the quote, add the `<small>` tag, and inside that, add the source's name wrapped in a `<cite>` tag. When you put it all together, you get something that looks like this:

That this is needed, desperately needed, is indicated by the incredible uptake of Bootstrap. I use it in all the server software I'm working on. And it shows through in the templating language I'm developing, so everyone who uses it will find it's "just there" and works, any time you want to do a Bootstrap technique. Nothing to do, no libraries to include. It's as if it were part of the hardware. Same approach that Apple took with the Mac OS in 1984.

— Developer of RSS, Dave Winer

Figure 2-7. Basic Blockquote

Blockquote Markup.

```
<blockquote>  
  <p>That this is needed, desperately needed, is indicated by the incredible uptake of Boots  
  <small>Developer of RSS, <cite title="Source Title">Dave Winer</cite></small>  
</blockquote>
```



If you want a `<blockquote>` with content that is right-aligned, simply add `.pull-right` to the tag. In addition to the right-aligned text, the entire blockquote is floated to the right. This creates nice pull-quotes in your content.

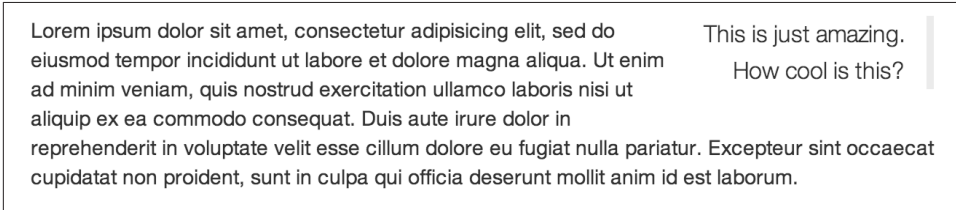


Figure 2-8. Pull-Right Blockquote

Lists

Bootstrap offers support and styling for the three main list types that HTML offers: ordered, unordered, and definition lists. An unordered list is a list that doesn't have any particular order, and is traditionally styled with bullets.

Unordered List

Unordered List Markup.

```
<h3>Favorite Outdoor Activites</h3>
<ul>
  <li>Backpacking in Yosemite</li>
  <li>Hiking in Arches
    <ul>
      <li>Delicate Arch</li>
      <li>Park Avenue</li>
    </ul>
  </li>
  <li>Biking the Flintstones Trail</li>
</ul>
```

Favorite Outdoor Activites

- Backpacking in Yosemite
- Hiking in Arches
 - Delicate Arch
 - Park Avenue
- Biking the Flintstones Trail

Figure 2-9. Unordered List Example

If you have an ordered list that you would like to remove the bullets from, add `class="unstyled"` to the opening `` tag.



Personally, I hold a strong aversion to using the `
` tag, and when I want a single spaced line break, I place each line in an unstyled unordered list. As an example, you might want a condensed address box, like the illustration above, you could have each line as a ``. In my mind, this is the more semantic way to markup the text.

Ordered List

An ordered list is a list that falls in some sort of sequential order, and is prefaced by numbers rather than bullets. This is handy when you want to build a list of numbered items, like a task list, guide items, or even a list of comments on a blog post.

Ordered List Markup.

```
<h3>Self-Referential Task List</h3>
<ol>
  <li>Turn off the internet.
  <li>Write the book</li>
  <li>... Profit?</li>
</ol>
```

Self-Referential Task List

1. Turn off the internet.
2. Right the book
3. ... Profit?

Figure 2-10. Ordered List Example

Definition List

The third type of list you get with Bootstrap is the definition list. The definition list differs from the ordered and unordered list in that instead of just having a block level `` element, 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 (or phrase) that is being defined. Subsequently, the `<dd>` is the definition of the `<dt>`.

A lot of times in markup, you will see people using headings inside an unordered list. This works, but maybe isn't the most semantic way to markup the text. A better idea would be to create a `<dl>` and then style the `<dt>` and `<dd>` as you would the heading and the text. That being said, out of the box, Bootstrap offers some clean default styles, and an option for a side-by-side layout of each definition.

Definition List Markup.

```
<h3>Common Electronics Parts</h3>
<dl>
  <dt>LED</dt>
  <dd>A light-emitting diode (LED) is a semiconductor light source.</dd>
  <dt>Servo</dt>
  <dd>Servos are small, cheap, mass-produced actuators used for radio control and small robots.</dd>
</dl>
```

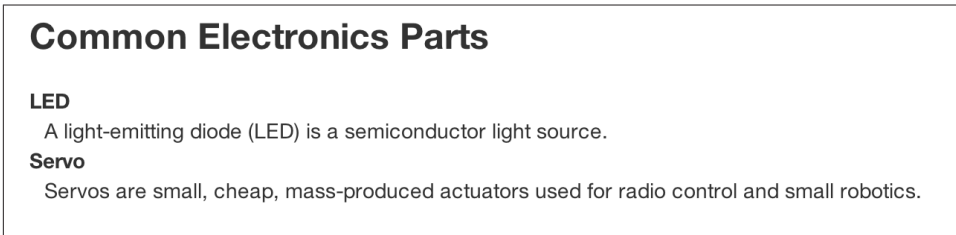


Figure 2-11. Definition List Example

To change the `<dl>` to a horizontal layout, with the `<dt>` on the left side, and the `<dd>` on the right, simply add `class="dl-horizontal"` to the opening tag.

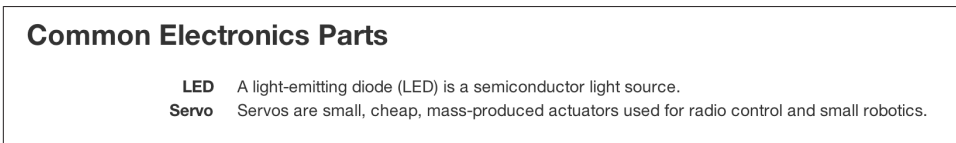


Figure 2-12. Horizontal Definition List Example



Horizontal description lists will truncate terms that are too long to fit in the left column with `text-overflow`. Additionally, in narrower viewports, they will automatically change to the default stacked layout.

Code

There are two different key ways to display code with Bootstrap. The first is the `<code>` tag, and the second is with the `<pre>` tag. Generally, if you are going to be displaying code inline, then you should use the `<code>` tag, but if it needs to be displayed as a standalone block element, or if it has multiple lines, then you should use the `<pre>` tag.

```
<p>Instead of always using divs, in HTML5, you can use new elements like <code>&lt;section&gt;</code>  
<pre>  
  &lt;article&gt;  
    &lt;h1&gt;Article Heading&lt;/h1&gt;  
  &lt;/article&gt;  
</pre>
```



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

Tables

One of my favorite parts of Bootstrap is the nice way that tables are handled. I do a lot of work looking at and building tables, and the clean layout is great feature that's included in Bootstrap right off the bat. Bootstrap supports the following elements:

Table 2-1. Table Elements Bootstrap Supports

Tag	Description
<code><table></code>	Wrapping element for displaying data in a tabular format
<code><thead></code>	Container element for table header rows (<code><tr></code>) to label table columns
<code><tbody></code>	Container element for table rows (<code><tr></code>) in the body of the table
<code><tr></code>	Container element for a set of table cells (<code><td></code> or <code><th></code>) that appears on a single row
<code><td></code>	Default table cell
<code><th></code>	Special table cell for column (or row, depending on scope and placement) labels. Must be used within a <code><thead></code>
<code><caption></code>	Description or summary of what the table holds, especially useful for screen readers

If you want a nice basic table style with just some light padding and horizontal dividers only, add the base class of `.table` to any table. The basic layout has a top border on all of the `<td>` elements.

Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 2-13. Basic Table Class

Table Base Class Example.

```
<table class="table">
  <caption>...</caption>
  <thead>
    <tr>
      <th>...</th>
      <th>...</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>...</td>
      <td>...</td>
    </tr>
  </tbody>
</table>
```

Optional Table Classes

With the base table markup, and adding the `.table` class, there are few additional classes that you can add to style the markup. There are three classes, `.table-striped`, `.table-bordered`, `.table-hover`, and `.table-condensed`.

Striped Table

By adding the `.table-striped` class, you will get stripes on rows within the `<tbody>`. This is done via the CSS `:nth-child` selector which is not available on Internet Explorer 7-8.

Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 2-14. Striped Table Class

Bordered Table

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

Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 2-15. Bordered Table Class

Hover Table

If you add the `.table-hover` class, when you hover over a row, a light grey background will be added to rows while the user hovers over them.

Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 2-16. Hover Table Class

Condensed Table

If you add the `.table-condensed` class, padding is cut in half on rows to condense the table. Useful if you want denser information.

Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 2-17. Condensed Table Class

Table Row Classes

If you want to style the table rows, you could add the following classes to change the background color.

Table 2-2. Optional Table Row Classes

Class	Description	Background Color
.success	Indicates a successful or positive action.	Green
.error	Indicates a dangerous or potentially negative action.	Red
.warning	Indicates a warning that might need attention.	Yellow
.info	Used as an alternative to the default styles.	Blue

#	Product	Payment Taken	Status
1	TB - Monthly	01/04/2012	Approved
2	TB - Monthly	02/04/2012	Declined
3	TB - Monthly	03/04/2012	Pending
4	TB - Monthly	04/04/2012	Call in to confirm

Figure 2-18. Table Row Classes Example

Forms

Another one of the highlights of using Bootstrap is the attention that is paid to forms. As a web developer, one of my least favorite things to do is style forms. Bootstrap makes it easy to do with the simple HTML markup and extended classes for different styles of forms.

The basic form structure comes styled in Bootstrap, without needing to add any extra helper classes. If you use the placeholder, it is only supported in newer browsers. In older ones, no text will be displayed.

Figure 2-19. Basic Form

Basic Form Structure.

```

<form>
  <fieldset>
    <legend>Legend</legend>
    <label for="name">Label name</label>
    <input type="text" id="name" placeholder="Type something...">
    <span class="help-block">Example block-level help text here.</span>
    <label class="checkbox" for="checkbox">
      <input type="checkbox" id="checkbox"> Check me out
    </label>
    <button type="submit" class="btn">Submit</button>
  </fieldset>
</form>

```

Optional Form Layouts

With a few helper classes, you can dynamically update the layout of your form. Bootstrap comes with a few preset styles you can use.

Search Form

Add `.form-search` to the `form` tag, and then `.search-query` to the `<input>` for an input box with rounded corners, and an inline submit button.

Basic Form Structure.

```

<form class="form-search">
  <input type="text" class="input-medium search-query">
  <button type="submit" class="btn">Search</button>
</form>

```



Figure 2-20. Search Form

Inline Form

To create a form where all of the elements are inline, and labels are along side, add the class `.form-inline` to the form tag. To have the label and the input on the same line, use the horizontal form below.

Inline Form Code.

```
<form class="form-inline">
  <input type="text" class="input-small" placeholder="Email">
  <input type="password" class="input-small" placeholder="Password">
  <label class="checkbox">
    <input type="checkbox" Remember me
  </label>
  <button type="submit" class="btn">Sign in</button>
</form>
```

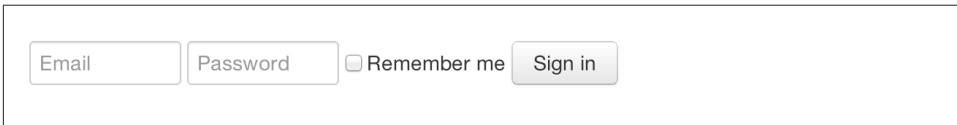


Figure 2-21. Inline Form Example

Horizontal Form

Bootstrap also comes with a pre-baked horizontal form; this one stands apart from the others not only in the amount of markup, but also in the presentation of the form. Traditionally you'd use a table to get a form layout like this, but Bootstrap manages to do it without. Even better, if you're using the responsive CSS, the horizontal form will automatically adapt to smaller layouts by stacking the controls vertically.

To create a form that uses the horizontal layout, do the following:

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

Figure 2-22. Horizontal Form Example

Horizontal Form Code.

```

<form class="form-horizontal">
  <div class="control-group">
    <label class="control-label" for="inputEmail">Email</label>
    <div class="controls">
      <input type="text" id="inputEmail" placeholder="Email">
    </div>
  </div>
  <div class="control-group">
    <label class="control-label" for="inputPassword">Password</label>
    <div class="controls">
      <input type="password" id="inputPassword" placeholder="Password">
    </div>
  </div>
  <div class="control-group">
    <div class="controls">
      <label class="checkbox">
        <input type="checkbox"> Remember me
      </label>
      <button type="submit" class="btn">Sign in</button>
    </div>
  </div>
</form>

```

Supported Form Controls

Bootstrap natively supports the most common form controls. Chief among them, input, textarea, checkbox and radio, and select.

Inputs

The most common form text field is the input—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.



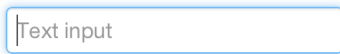
Figure 2-23. Input Example

Input Code.

```
<input type="text" placeholder="Text input">
```



Both `input` and `textarea` default to a nice blue glow when in the `:active` state.



Textarea

The `textarea` is used when you need multiple lines of input. You'll find you mainly modify the `rows` attribute, changing it to the number of rows that you need to support (fewer rows = smaller box, more rows = bigger box).

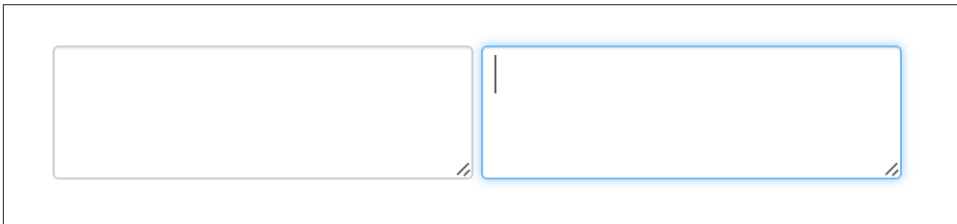


Figure 2-24. Both the `:active`, and the default `textarea`

Textarea Example.

```
<textarea rows="3"></textarea>
```

Checkboxes and radios

Checkboxes and radio buttons are great for when you want users to be able 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, and `radio` if you want to limit them to just one selection.

- Option one is this and that—be sure to include why it's great
- Option one is this and that—be sure to include why it's great
- Option two can be something else and selecting it will deselect option one

Figure 2-25. Radio and Checkbox example

Radio and Checkbox Code Example.

```

<label class="checkbox">
  <input type="checkbox" value=""
  Option one is this and that—be sure to include why it's great
</label>

<label class="radio">
  <input type="radio" name="optionsRadios" id="optionsRadios1" value="option1" checked>
  Option one is this and that—be sure to include why it's great
</label>
<label class="radio">
  <input type="radio" name="optionsRadios" id="optionsRadios2" value="option2">
  Option two can be something else and selecting it will deselect option one
</label>

```

- Option one is this and that—be sure to include why it's great
- Option one is this and that—be sure to include why it's great
- Option two can be something else and selecting it will deselect option one

Figure 2-26. Inline Checkboxes

If you want multiple checkboxes to appear on the same line together, simply add the `.inline` class to a series of checkboxes or radios.

```

<label for="option1" class="checkbox inline">
  <input id="option1" type="checkbox" id="inlineCheckbox1" value="option1"> 1
</label>
<label for="option2" class="checkbox inline">
  <input id="option2" type="checkbox" id="inlineCheckbox2" value="option2"> 2
</label>
<label for="option3" class="checkbox inline">
  <input id="option3" type="checkbox" id="inlineCheckbox3" value="option3"> 3
</label>

```

Selects

A select is used when you want to allow the user to pick from multiple options, but by default it only allows one. It's best to use `<select>` for list options of which the user is familiar such as states or numbers. Use `multiple="multiple"` to allow the user to select more than one option. If you only want the user to choose one option, use `type="radio"`.



Figure 2-27. Select Example

Select Code Example.

```
<select>
  <option>1</option>
  <option>2</option>
  <option>3</option>
  <option>4</option>
  <option>5</option>
</select>

<select multiple="multiple">
  <option>1</option>
  <option>2</option>
  <option>3</option>
  <option>4</option>
  <option>5</option>
</select>
```

Extending Form Controls

In addition to the basic form controls listed in the previous section, Bootstrap offers a few other form components to complement the standard HTML form elements; for example, it lets you easily prepend and append content to inputs.

Prepended and Appended Inputs

By adding prepended and appended content to an input field, you can add common elements to the text users input, like the dollar symbol, the @ for a Twitter username or anything else that might be common for your application interface. To use, wrap the input in a div with class `input-prepend` (to add the extra content before the user input)

or `input-append` (to add it after). Then, within that same `<div>`, place your extra content inside a `` with an add-on class, and place the `` either before or after the `<input>` element.



Figure 2-28. Prepend and Append Example

Prepend and Append Code Example.

```
<div class="input-prepend">
  <span class="add-on">@</span>
  <input class="span2" id="prependedInput" type="text" placeholder="Username">
</div>
<div class="input-append">
  <input class="span2" id="appendedInput" type="text">
  <span class="add-on">.00</span>
</div>
```

If you combine both of them, you simply need to add both the `.input-prepend` and `.input-append` classes to the parent `<div>`.



Figure 2-29. Using both the append and prepend

Append and Prepend Code Example.

```
<div class="input-prepend input-append">
  <span class="add-on">${</span>
  <input class="span2" id="appendedPrependedInput" type="text">
  <span class="add-on">.00</span>
</div>
```

Rather than using a ``, you can instead use `<button>` with a class of `btn` to attach (surprise!) a button or two to the input.

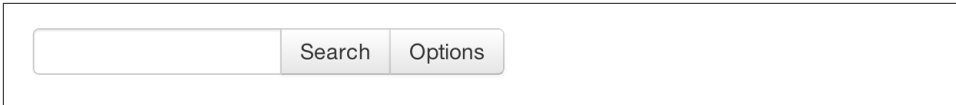


Figure 2-30. Attach multiple buttons to an input

Attach Multiple Buttons Code Example.

```
<div class="input-append">
  <input class="span2" id="appendedInputButtons" type="text">
  <button class="btn" type="button">Search</button>
  <button class="btn" type="button">Options</button>
</div>
```

If you are appending a button to a search form, you will get the same nice rounded corners that you would expect.

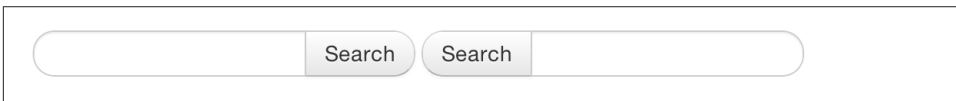


Figure 2-31. Append Button to Search Form

```
<form class="form-search">
  <div class="input-append">
    <input type="text" class="span2 search-query">
    <button type="submit" class="btn">Search</button>
  </div>
  <div class="input-prepend">
    <button type="submit" class="btn">Search</button>
    <input type="text" class="span2 search-query">
  </div>
</form>
```

Form Control Sizing

With the default grid system that is inherent in Bootstrap, you can use the `.span*` system for sizing form controls. In addition to the span column-sizing method, you can also use a handful of classes that take a relative approach to sizing. If you want the input to act as a block level element, you can add `.input-block-level` and it will be the full width of the container element.

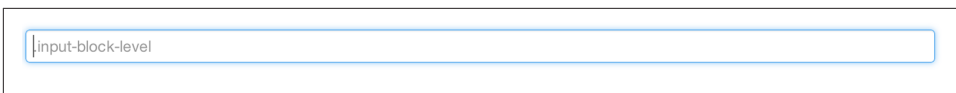


Figure 2-32. Block Level Input

```
<input class="input-block-level" type="text" placeholder=".input-block-level">
```

Relative Input Controls

```
<input class="input-mini" type="text" placeholder=".input-mini">  
<input class="input-small" type="text" placeholder=".input-small">  
<input class="input-medium" type="text" placeholder=".input-medium">  
<input class="input-large" type="text" placeholder=".input-large">  
<input class="input-xlarge" type="text" placeholder=".input-xlarge">  
<input class="input-xxlarge" type="text" placeholder=".input-xxlarge">
```



In future versions of Bootstrap, these input classes will be altered to match the button sizes. For example, `.input-large` will increase the padding and font-size of an input.

Grid Sizing

You can use any `.span` from `.span1` to `.span12` for form control sizing.

```

<input class="span1" type="text" placeholder=".span1">
<input class="span2" type="text" placeholder=".span2">
<input class="span3" type="text" placeholder=".span3">
<select class="span1">
  ...
</select>
<select class="span2">
  ...
</select>
<select class="span3">
  ...
</select>

```

If you want to use multiple inputs on a line, simply use the `.controls-row` modifier class to apply the proper spacing. It floats the inputs to collapse the white space, and set the correct margins, and like the `.row` class, it also clears the float.



Figure 2-33. Control Row

```

<div class="controls">
  <input class="span5" type="text" placeholder=".span5">
</div>
<div class="controls controls-row">
  <input class="span4" type="text" placeholder=".span4">
  <input class="span1" type="text" placeholder=".span1">
</div>
...

```

Uneditable Text

If you want to present a form control, but not have it editable, simple add the class `.uneditable-input`.



Figure 2-34. Uneditable Input

```
<span class="input-xlarge uneditable-input">Some value here</span>
```

Form Actions

At the bottom of a horizontal-form you can place the form actions. Then inputs will correctly line up with the floated form controls.



Figure 2-35. Form Controls

```
<div class="form-actions">  
  <button type="submit" class="btn btn-primary">Save changes</button>  
  <button type="button" class="btn">Cancel</button>  
</div>
```

Help Text

Bootstrap form controls can have either block or inline text that flows with the inputs.

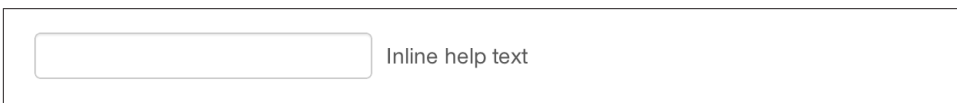


Figure 2-36. Inline Help

```
<input type="text"><span class="help-inline">Inline help text</span>
```

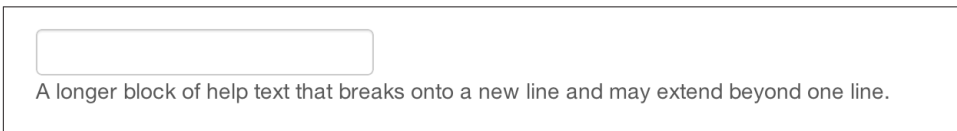


Figure 2-37. Block Help

```
<input type="text"><span class="help-block">A longer block of help text that breaks onto a new line
```


Form Control States

In addition to the `:focus` state, Bootstrap offers styling for disabled inputs, and classes for form validation.

Input Focus

When an input receives `:focus`, that is to say, a user clicks into the input, or tabs into it, the outline of the input is removed, and a box-shadow is applied. I remember the first time that I saw this on Twitter's site, it blew me away, and I had to dig into the code to see how they did it. In WebKit, this accomplished in the following manner:

```
input {
  -webkit-box-shadow: inset 0 1px 1px rgba(0, 0, 0, 0.075);
  -webkit-transition: box-shadow linear 0.2s;
}

input:focus {
  -webkit-box-shadow: inset 0 1px 1px rgba(0, 0, 0, 0.075), 0 0 8px rgba(82, 168, 236, 0.6);
}
```

The `<input>` has a small inset box-shadow, this gives the appearance that the input sits lower than the page. When `:focus` is applied, an 8px light blue code is applied. The `webkit-transition` tells the browser to apply the effect in a linear manner over 0.2 seconds. Nice and subtle, a great effect.



Figure 2-38. Focused Input

```
<input class="input-xlarge" id="focusedInput" type="text" value="This is focused...">
```

Disabled Input

If you need to disable an input, simply add the `disabled` attribute to not only disable it, but change the styling, and the mouse cursor when it hover over the element.



Figure 2-39. Disabled Input

```
<input class="input-xlarge" id="disabledInput" type="text" placeholder="Disabled input here..." disabled="">
```

Validation States

Bootstrap includes validation styles for error, warning, info, and success messages. To use, simply add the appropriate class to the surrounding `.control-group`.

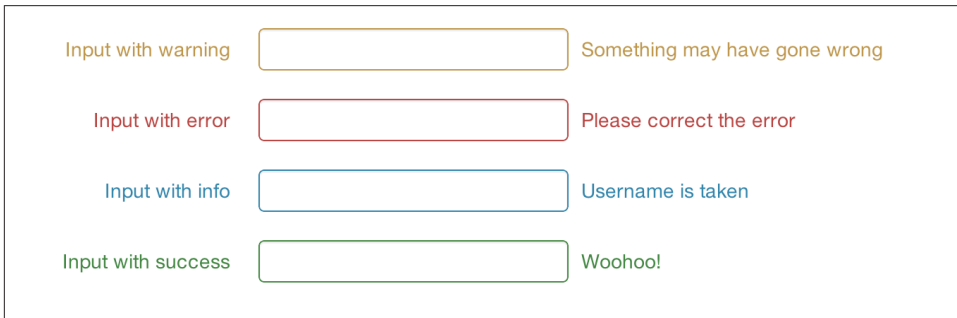


Figure 2-40. Validation States

```
<div class="control-group warning">
  <label class="control-label" for="inputWarning">Input with warning</label>
  <div class="controls">
    <input type="text" id="inputWarning">
    <span class="help-inline">Something may have gone wrong</span>
  </div>
</div>
<div class="control-group error">
  <label class="control-label" for="inputError">Input with error</label>
  <div class="controls">
    <input type="text" id="inputError">
    <span class="help-inline">Please correct the error</span>
  </div>
</div>
<div class="control-group success">
  <label class="control-label" for="inputSuccess">Input with success</label>
  <div class="controls">
    <input type="text" id="inputSuccess">
    <span class="help-inline">Woohoo!</span>
  </div>
</div>
```

Buttons

One of my favorite features of Bootstrap is the way that buttons are styled. Dave Winer, inventor of RSS, and big fan of Bootstrap has this to say about it:

That this is needed, desperately needed, is indicated by the incredible uptake of Bootstrap. I use it in all the server software I'm working on. And it shows through in the templating language I'm developing, so everyone who uses it will find it's "just there" and works, any









time you want to do a Bootstrap technique. Nothing to do, no libraries to include. It's as if it were part of the hardware. Same approach that Apple took with the Mac OS in 1984.

— Dave Winer
scripting.com

I like to think that Bootstrap is doing that, unifying the web, and allowing a unified experience of what an interface can look like across the web. With the advent of Bootstrap, you can spot the sites that have taken it up usually first by the buttons that they use. A grid layout, and many of the other features fade into the background, but buttons, forms and other unifying elements are a key part of Bootstrap. Maybe I'm the only person that does this, but when I come across a site that is using Bootstrap, I want to give a high five to whomever answers the webmaster email at that domain, as they probably just get it. It reminds me of a few years ago I would do the same thing when I would see wp-content in the HTML of sites that I would visit.

Now, buttons, and links can all look alike with Bootstrap, anything that is given that class of `btn` will inherit the default look of a grey button with rounded corners. Adding extra classes will add colors to the buttons.

Table 2-3. Button Color Examples

Buttons	Class	Description
	<code>btn</code>	Standard gray button with gradient
	<code>btn btn-primary</code>	Provides extra visual weight and identifies the primary action in a set of buttons
	<code>btn btn-info</code>	Used as an alternative to the default styles
	<code>btn-success</code>	Indicates a successful or positive action
	<code>btn btn-warning</code>	Standard gray button with gradient
	<code>btn btn-danger</code>	Indicates a dangerous or potentially negative action
	<code>btn btn-inverse</code>	Alternate dark gray button, not tied to a semantic action or use
	<code>btn btn-link</code>	Deemphasize a button by making it look like a link while maintaining button behavior



There are issues with Internet Explorer 9 not cropping background gradients on rounded corners, so buttons aren't shown. Also, Internet Explorer doesn't work well with disabled button elements. The rendered text is gray with a nasty text-shadow that hasn't been fixed.

Button Sizes

If you need larger or smaller buttons, simply add `.btn-large`, `.btn-small`, or `.btn-mini` to links or buttons.



Figure 2-41. Different Button Sizes

```
<p>
  <button class="btn btn-large btn-primary" type="button">Large button</button>
  <button class="btn btn-large" type="button">Large button</button>
</p>
<p>
  <button class="btn btn-primary" type="button">Default button</button>
  <button class="btn" type="button">Default button</button>
</p>
<p>
  <button class="btn btn-small btn-primary" type="button">Small button</button>
  <button class="btn btn-small" type="button">Small button</button>
</p>
<p>
  <button class="btn btn-mini btn-primary" type="button">Mini button</button>
  <button class="btn btn-mini" type="button">Mini button</button>
</p>
```

If you want to create buttons that display like a block level element, simply add the `btn-block` class. These buttons will display at 100% width.

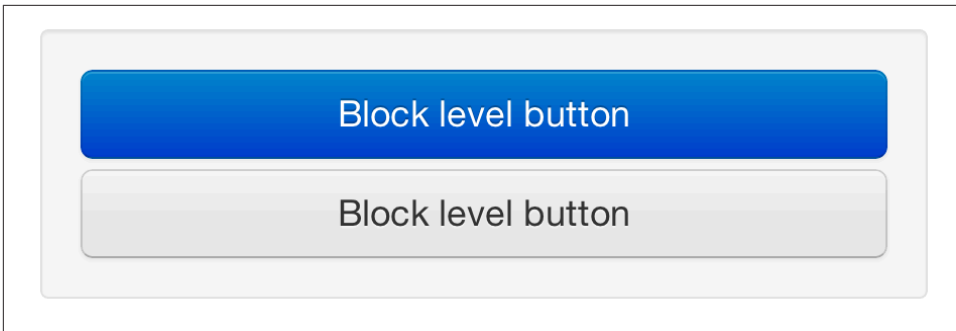


Figure 2-42. Block Level Button

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

Disabled Button Styling

For anchor elements, simply add the class of `.disabled` to the tag, and the link will drop back in color, and will lose the gradient.



The `.disabled` class is being used much like the `.active` class. So, no `.btn` prefix, and remember, this is only for looks, to truly disable the link, you will want to use some javascript to really disable the link.



Figure 2-43. Disabled Link

```
<a href="#" class="btn btn-large btn-primary disabled">Primary link</a>  
<a href="#" class="btn btn-large disabled">Link</a>
```

For a button, simply add the `disabled` attribute to the button. This will actually disable the button, so javascript is not directly needed.



Figure 2-44. Disabled Button

```
<button type="button" class="btn btn-large btn-primary disabled" disabled="disabled">Primary button</button>  
<button type="button" class="btn btn-large" disabled>Button</button>
```

Images

Images have three classes to apply some simple styles. They are `.img-rounded` that adds `border-radius:6px` to give the image rounded corners, `.img-circle` that adds makes the entire image a circle by adding `border-radius:500px` making the image round, and lastly, `img-polaroid`, that adds a bit of padding and a grey border.



Figure 2-45. Images

```
  
  

```

Icons

Bootstrap bundles 140 icons into one sprite that can be used with buttons, links, navigation, and and form fields. The icons are provided by [Glyphicons](#).

 icon-glass	 icon-music	 icon-search	 icon-envelope
 icon-heart	 icon-star	 icon-star-empty	 icon-user
 icon-film	 icon-th-large	 icon-th	 icon-th-list
 icon-ok	 icon-remove	 icon-zoom-in	 icon-zoom-out
 icon-off	 icon-signal	 icon-cog	 icon-trash
 icon-home	 icon-file	 icon-time	 icon-road
 icon-download-alt	 icon-download	 icon-upload	 icon-inbox
 icon-play-circle	 icon-repeat	 icon-refresh	 icon-list-alt
 icon-lock	 icon-flag	 icon-headphones	 icon-volume-off
 icon-volume-down	 icon-volume-up	 icon-qrcode	 icon-barcode
 icon-tag	 icon-tags	 icon-book	 icon-bookmark
 icon-print	 icon-camera	 icon-font	 icon-bold
 icon-italic	 icon-text-height	 icon-text-width	 icon-align-left
 icon-align-center	 icon-align-right	 icon-align-justify	 icon-list
 icon-indent-left	 icon-indent-right	 icon-facetime-video	 icon-picture
 icon-pencil	 icon-map-marker	 icon-adjust	 icon-tint
 icon-edit	 icon-share	 icon-check	 icon-move
 icon-step-backward	 icon-fast-backward	 icon-backward	 icon-play
 icon-pause	 icon-stop	 icon-forward	 icon-fast-forward
 icon-step-forward	 icon-eject	 icon-chevron-left	 icon-chevron-right
 icon-plus-sign	 icon-minus-sign	 icon-remove-sign	 icon-ok-sign
 icon-question-sign	 icon-info-sign	 icon-screenshot	 icon-remove-circle
 icon-ok-circle	 icon-ban-circle	 icon-arrow-left	 icon-arrow-right
 icon-arrow-up	 icon-arrow-down	 icon-share-alt	 icon-resize-full
 icon-resize-small	 icon-plus	 icon-minus	 icon-asterisk
 icon-exclamation-sign	 icon-gift	 icon-leaf	 icon-fire
 icon-eye-open	 icon-eye-close	 icon-warning-sign	 icon-plane
 icon-calendar	 icon-random	 icon-comment	 icon-magnet
 icon-chevron-up	 icon-chevron-down	 icon-retweet	 icon-shopping-cart
 icon-folder-close	 icon-folder-open	 icon-resize-vertical	 icon-resize-horizontal
 icon-hdd	 icon-bullhorn	 icon-bell	 icon-certificate
 icon-thumbs-up	 icon-thumbs-down	 icon-hand-right	 icon-hand-left
 icon-hand-up	 icon-hand-down	 icon-circle-arrow-right	 icon-circle-arrow-left
 icon-circle-arrow-up	 icon-circle-arrow-down	 icon-globe	 icon-wrench
 icon-tasks	 icon-filter	 icon-briefcase	 icon-fullscreen

Glyphicon Attribution

Users of Bootstrap are fortunate to use the Glyphicons free of use on Bootstrap projects. The developers have asked that you use a link back to Glyphicons when practical.

Glyphicons Halflings are normally not available for free, but an arrangement between Bootstrap and the Glyphicons creators have made this possible at no cost to you as de-

velopers. As a thank you, we ask you to include an optional link back to Glyphicons whenever practical.

— Bootstrap Documentation
<http://getbootstrap.com>

Usage

To use the icons, simply use an `<i>` tag with the namespaced `.icon-` class. For example, if you wanted to use the edit icon, you would simply add the `.icon-edit` class to the `<i>` tag.

```
<i class="icon-edit"></i>
```

If you want to use the white icon, simply add the `.icon-white` class to the tag.

```
<i class="icon-edit icon-white"></i>
```

Button Groups

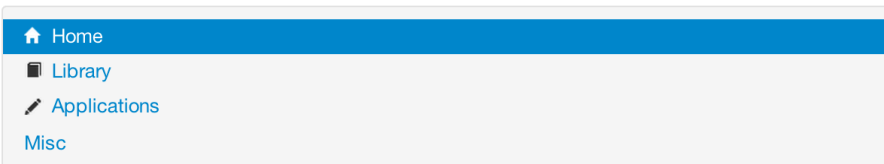
Using button groups, combined with icons, you can create nice interface elements with minimal markup.



```
<div class="btn-toolbar">
  <div class="btn-group">
    <a class="btn" href="#"><i class="icon-align-left"></i></a>
    <a class="btn" href="#"><i class="icon-align-center"></i></a>
    <a class="btn" href="#"><i class="icon-align-right"></i></a>
    <a class="btn" href="#"><i class="icon-align-justify"></i></a>
  </div>
</div>
```

Navigation

When you are using icons next to a string of text, make sure to add a space to provide the proper alignment of the image. More of navigation code will be covered in the next chapter.



```
<ul class="nav nav-list">
  <li class="active"><a href="#"><i class="icon-home icon-white"></i> Home</a></li>
  <li><a href="#"><i class="icon-book"></i> Library</a></li>
```



```
<li><a href="#"><i class="icon-pencil"></i> Applications</a></li>
<li><a href="#"><i class="i"></i> Misc</a></li>
</ul>
```

Bootstrap Layout Components

In addition to all of the markup provided in the previous chapter, Bootstrap provides a toolkit of flexible components that can be used in designing application interfaces, web features and more. Each of the plugins are available in a separate Javascript file, combined all together, or, using the Bootstrap customizer, you can pick and choose which plugins you want. Personally, on the projects that I build, I lump them all together. That way I have options.

Dropdown Menus

Dropdown menus are a toggleable, contextual menu for displaying links in a list format. The dropdowns can be used on a variety of different elements, navs, buttons, and more. You can have a single dropdown, or extend the dropdown into another sub-menu.



Figure 3-1. Basic Dropdown Menu

```
<ul class="dropdown-menu" role="menu" aria-labelledby="dropdownMenu">
  <li><a tabindex="-1" href="#">Action</a></li>
  <li><a tabindex="-1" href="#">Another action</a></li>
  <li><a tabindex="-1" href="#">Something else here</a></li>
  <li class="divider"></li>
```

```
<li><a tabindex="-1" href="#">Separated link</a></li>
</ul>
```

Options

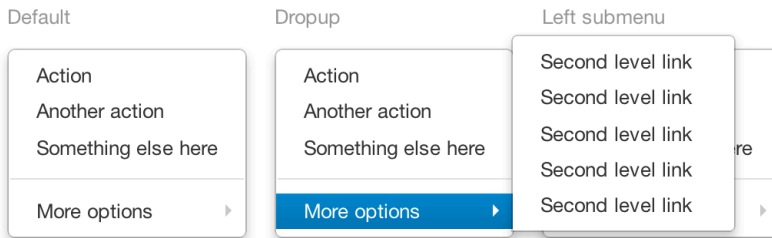
Right align the dropdown

Add `.pull-right` to a `.dropdown-menu` to right-align the dropdown menu to the parent object.

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

If you would like to add a second layer of dropdowns, simply add `.dropdown-submenu` to any `` in an existing dropdown menu for automatic styling.

Dropdown Submenu



```
<ul class="dropdown-menu" role="menu" aria-labelledby="dLabel">
  ...
  <li class="dropdown-submenu">
    <a tabindex="-1" href="#">More options</a>
    <ul class="dropdown-menu">
      ...
    </ul>
  </li>
</ul>
```

Button Groups

Button groups allow multiple buttons to be stacked together. This is useful like the example below when you want to place items like alignment buttons together. Simply, to create a button group, simply wrap a series of anchors or buttons with a class of `.btn` with a `<div>` that has `.btn-group` as a class.



Figure 3-2. Inline Button Group

Inline Button Group Code Example.

```
<div class="btn-group">
  <button class="btn">1</button>
  <button class="btn">2</button>
  <button class="btn">3</button>
</div>
```

If you have multiple button groups that you want to align on a single line, wrap multiple `.btn-group` with `.btn-toolbar`. For more information about using icons with buttons, follow the examples in chapter 2.

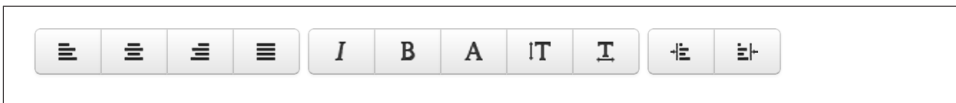


Figure 3-3. Button Toolbar

Button Toolbar Code Example.

```
<div class="btn-toolbar">
  <div class="btn-group">
    <a class="btn" href="#"><i class="icon-align-left"></i></a>
    <a class="btn" href="#"><i class="icon-align-center"></i></a>
    <a class="btn" href="#"><i class="icon-align-right"></i></a>
    <a class="btn" href="#"><i class="icon-align-justify"></i></a>
  </div>
  <div class="btn-group">
    <a class="btn" href="#"><i class="icon-italic"></i></a>
    <a class="btn" href="#"><i class="icon-bold"></i></a>
    <a class="btn" href="#"><i class="icon-font"></i></a>
    <a class="btn" href="#"><i class="icon-text-height"></i></a>
    <a class="btn" href="#"><i class="icon-text-width"></i></a>
  </div>
  <div class="btn-group">
    <a class="btn" href="#"><i class="icon-indent-left"></i></a>
    <a class="btn" href="#"><i class="icon-indent-right"></i></a>
  </div>
</div>
```

To make the buttons stack, simply add `.btn-group-vertical` to the `.btn-group` class.

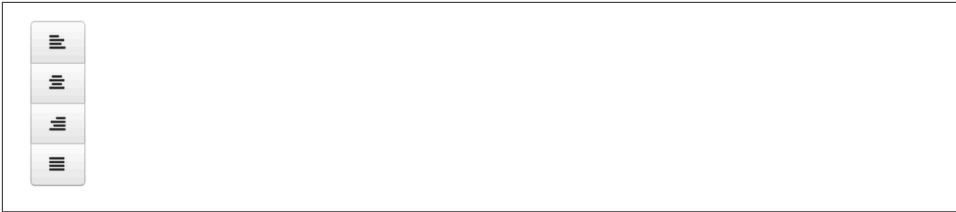


Figure 3-4. Vertical Button Group

Vertical Button Group Code.

```
<div class="btn-group btn-group-vertical">  
  ...  
</div>
```

Button Groups as Radio and Checkboxes

To have the checkboxes function as radio buttons where only one can be selected at a time, or checkboxes where multiple can be selected, you simply need to add some extra markup and the Bootstrap JavaScript will provide the rest. This will be covered in detail in chapter 4.



To use a button with a dropdown, they must be individually wrapped in their own `btn-group` within a `btn-toolbar` for proper rendering.

Buttons With Dropdowns

To add a dropdown to a button, simply wrap a button and a dropdown menu in a `.btn-group`. You can also use a `` to act as an indicator that the button is a dropdown.

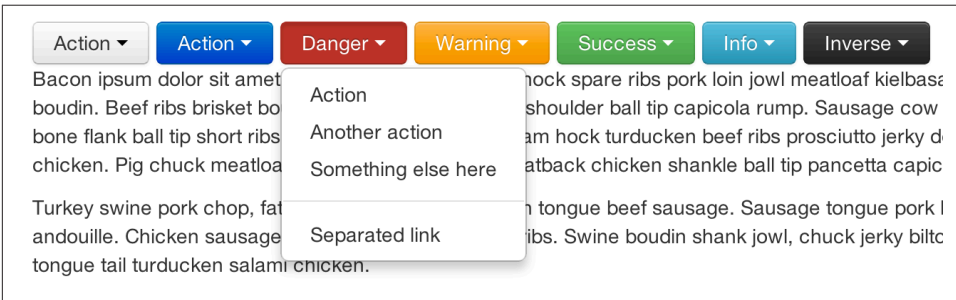


Figure 3-5. Button with a Dropdown

Button Dropdown Code.

```
<div class="btn-group">
  <button class="btn btn-danger">Danger</button>
  <button class="btn btn-danger dropdown-toggle" data-toggle="dropdown">
    <span class="caret"></span>
  </button>
  <ul class="dropdown-menu">
    <li><a href="#">Action</a></li>
    <li><a href="#">Another action</a></li>
    <li><a href="#">Something else here</a></li>
    <li class="divider"></li>
    <li><a href="#">Separated link</a></li>
  </ul>
</div>
```

You can use the dropdowns with any button size, `.btn-large` `.btn`, `.btn-small` and `.btn-mini`.



Figure 3-6. Button Dropdown Sizes

Split Button Dropdowns

Using the same general styles of the dropdown button, but adding a primary action along with the dropdown, split buttons have the primary action on the left, and the a toggle on the right for the dropdown.

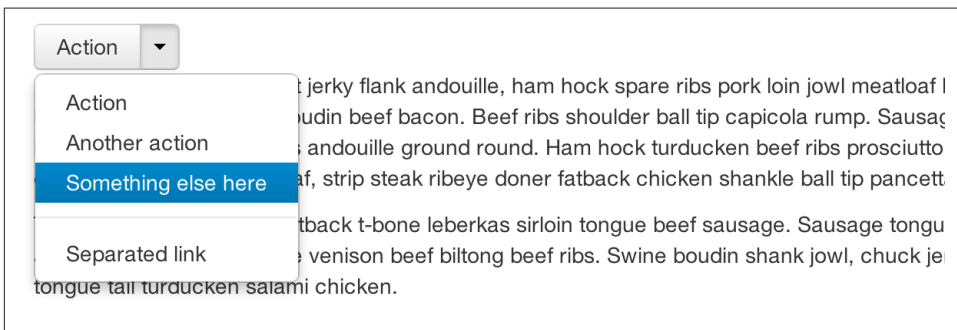


Figure 3-7. Split Button Dropdown

Split Button Dropdown Code Example.

```

<div class="btn-group">
  <button class="btn">Action</button>
  <button class="btn dropdown-toggle" data-toggle="dropdown">
    <span class="caret"></span>
  </button>
  <ul class="dropdown-menu">
    <!-- dropdown menu links -->
  </ul>
</div>

```

<http://jsfiddle.net/vSHkJ/embedded/result/>

Dropup Menu

Menus can also be built to dropup, rather than down. To make this change, simply add `.dropup` to the `.btn-group` container. To have the button pullup from the right hand side, add `.pull-right` to the `.dropdown-menu`. Take notice, the caret is now pointed up, as the menu will be going up instead of down.



Figure 3-8. Dropup Menu

Dropup Menu Code Example.

```

<div class="btn-group dropup">
  <button class="btn">Dropup</button>
  <button class="btn dropdown-toggle" data-toggle="dropdown">
    <span class="caret"></span>
  </button>
  <ul class="dropdown-menu">
    <!-- dropdown menu links -->
  </ul>
</div>

```

<http://jsfiddle.net/DyfSZ/3/embedded/result/>

Navigation Elements

Bootstrap provides a few different opportunities for styling navigation elements. All of them share the same markup and base class `.nav`.

Bootstrap also provides a helper class, `.active`. In principal, it generally adds distinction to the current element, and sets it apart from the the rest of the navigation elements. You could add this class to the home page links, or add it to the links of the page that you are currently on.

Tabular Navigation

To create a tabbed navigation menu, start with a basic unordered list with the base class of `.nav` and add `.nav-tabs`.



Figure 3-9. Tabbed Navigation

Tabbed Navigation Code Example.

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

Basic Pills Navigation

To turn the tabs into pills, instead of using the `.nav-tabs` use `.nav-pills`.



Figure 3-10. Tabbed Navigation

Tabbed Navigation Code Example.

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

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

Disabled Class

For each of the `.nav` classes, if you add the `.disabled` class, it will create gray link that also disables the hover state. The link is still clickable unless the `href` is removed, with javascript or some other method.



Figure 3-11. Disabled Navigation

Disabled Navigation Code Example.

```
<ul class="nav nav-pills">
  ...
  <li class="disabled"><a href="#">Home</a></li>
  ...
</ul>
```

Stackable Navs

Both tabs and pills are horizontal by default, to make them stackable, just add the `.nav-stacked` class to make them appear vertically stacked.



Figure 3-12. Stacked Tabs

Stacked Tabs Code Example.

```
<ul class="nav nav-tabs nav-stacked">
  ...
</ul>
```

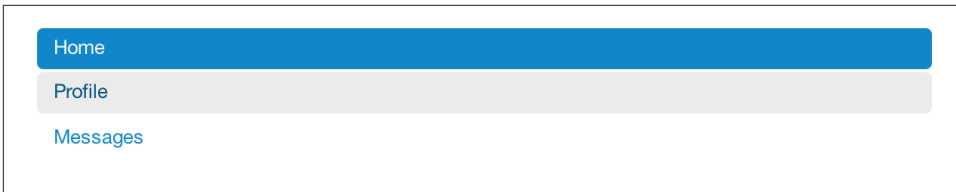


Figure 3-13. Stacked Pills

Stacked Pills Code Example.

```
<ul class="nav nav-pills nav-stacked">
  ...
</ul>
```

Dropdowns

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

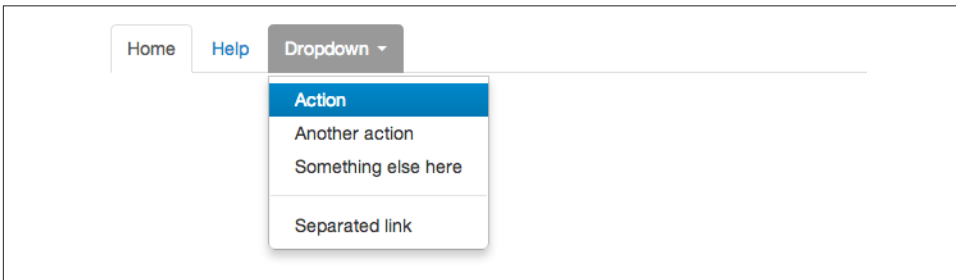


Figure 3-14. Tabbed Navigation with a Dropdown Menu

Tabbed Navigation Dropdown Code Example.

```
<ul class="nav nav-tabs">
  <li class="dropdown">
    <a class="dropdown-toggle"
      data-toggle="dropdown"
      href="#">
      Dropdown
    <b class="caret"></b>
    </a>
    <ul class="dropdown-menu">
      <li><a href="#">Action</a></li>
      <li><a href="#">Another action</a></li>
      <li><a href="#">Something else here</a></li>
      <li class="divider"></li>
      <li><a href="#">Separated link</a></li>
    </ul>
  </li>
</ul>
```

```

    </ul>
  </li>
</ul>

```

To do the same thing with pills, simply swap the `.nav-tabs` class with `.nav-pills`.

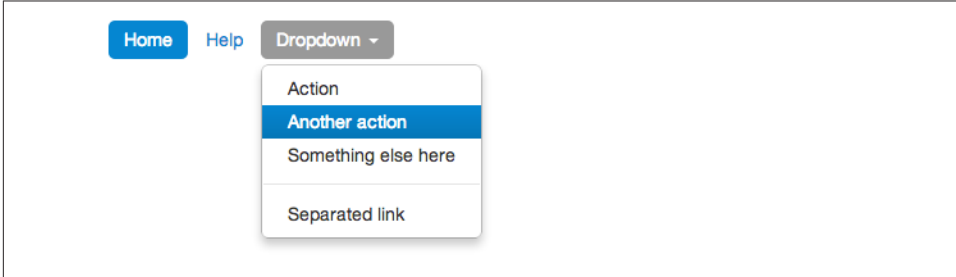


Figure 3-15. Pill Navigation with Dropdowns

Pill Navigation Dropdown Code Example.

```

<ul class="nav nav-pills">
  <li class="dropdown">
    <a class="dropdown-toggle"
      data-toggle="dropdown"
      href="#">
      Dropdown
    <b class="caret"></b>
    </a>
    <ul class="dropdown-menu">
      <!-- links -->
    </ul>
  </li>
</ul>

```

Navigation Lists

Navigation lists are useful when you need to display a group of navigation links. This type of interface element is common when building admin interfaces in websites. In the MAKE admin interface, I have one of these on the sidebar of every page with quick links to common pages. A form of this is what that Bootstrap developers use for their documentation. Like all of the lists that we have discussed thus far, this is simply an unordered list with the `.nav` class, and to give it its specific styling, we add the `.nav-list` class.



Figure 3-16. Navigation List

Navigation List Code Example.

```

<ul class="nav nav-list">
  <li class="nav-header">List Header</li>
  <li class="active"><a href="/">Home</a></li>
  <li><a href="#">Blog</a></li>
  <li class="divider"></li>
  <li><a href="#">Contact</a></li>
</ul>

```

Horizontal Divider

To create a divider, much like an `<hr />`, use an empty `` with a class of `.divider`.

Horizontal Divider.

```

<ul class="nav-menu">
  ...
  <li class="divider"></li>
  ...
</ul>

```

Tabbable Navigation

Not only can you create a tabbed navigation, but by using the JavaScript plugin, you can also add some interaction by making them tab able to open different windows of content. To make navigation tabs tabbable, create a `.tab-pane` with a unique ID for every tab, and then wrap them in `.tab-content`.

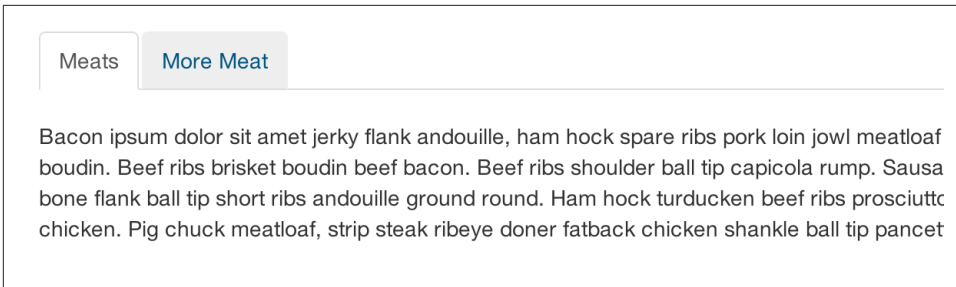


Figure 3-17. *Tabbable Navigation Example*

Table Navigation Code Example.

```

<div class="tabbable">
  <ul class="nav nav-tabs">
    <li class="active"><a href="#tab1" data-toggle="tab">Meats</a></li>
    <li><a href="#tab2" data-toggle="tab">More Meats</a></li>
  </ul>
  <div class="tab-content">
    <div class="tab-pane active" id="tab1">
      <p>Bacon ipsum dolor sit amet jerky flank...</p>
    </div>
    <div class="tab-pane" id="tab2">
      <p>Beef ribs, turducken ham hock...</p>
    </div>
  </div>
</div>

```

<http://jsfiddle.net/JUqAT/embedded/result/>

If you want to make the tabs fade when switching, add `.fade` to each `.tab-pane`.

Tab Position

The tabs are fully positionable, you can have them above, below, or on the sides of the content.



Figure 3-18. *Bottom Tabs*

Bottom Tab Code Example.

```

<div class="tabbable tabs-below">
  <div class="tab-content">

```

```

<div class="tab-pane active" id="tab1">
  <p>I'm in section A.</p>
</div>
<div class="tab-pane" id="tab2">
  <p>I'm in section B.</p>
</div>
<div class="tab-pane" id="tab3">
  <p>I'm in section C.</p>
</div>
</div>
<ul class="nav nav-tabs">
  <li class="active"><a href="#tab1" data-toggle="tab">Section 1</a></li>
  <li><a href="#tab2" data-toggle="tab">Section 2</a></li>
  <li><a href="#tab3" data-toggle="tab">Section 3</a></li>
</ul>
</div>

```

Tabs on the left get the `.tabs-left` class.



Figure 3-19. Left Tabs

Left Tab Code Example.

```

<div class="tabbable tabs-left">
  <div class="tab-content">
    <div class="tab-pane active" id="tab1">
      <p>I'm in section A.</p>
    </div>
    <div class="tab-pane" id="tab2">
      <p>I'm in section B.</p>
    </div>
    <div class="tab-pane" id="tab3">
      <p>I'm in section C.</p>
    </div>
  </div>
  <ul class="nav nav-tabs">
    <li class="active"><a href="#tab1" data-toggle="tab">Section 1</a></li>
    <li><a href="#tab2" data-toggle="tab">Section 2</a></li>
    <li><a href="#tab3" data-toggle="tab">Section 3</a></li>
  </ul>
</div>

```

Tabs on the right get the `.tabs-right` class.



Figure 3-20. Right Tabs

Right Tab Code Example.

```

<div class="tabbable tabs-right">
  <div class="tab-content">
    <div class="tab-pane active" id="tab1">
      <p>I'm in section A.</p>
    </div>
    <div class="tab-pane" id="tab2">
      <p>I'm in section B.</p>
    </div>
    <div class="tab-pane" id="tab3">
      <p>I'm in section C.</p>
    </div>
  </div>
  <ul class="nav nav-tabs">
    <li class="active"><a href="#tab1" data-toggle="tab">Section 1</a></li>
    <li><a href="#tab2" data-toggle="tab">Section 2</a></li>
    <li><a href="#tab3" data-toggle="tab">Section 3</a></li>
  </ul>
</div>

```



As a footnote here to the tabbable elements, you can use the markup here to control a variety of things, perhaps outside of the scope of the default usage mechanism. On MAKE's site, I used this to control the navigation, and subnavigation. When you click on the navigation menu, the sub navigation would change and show different links.

Navbar

The Navbar is a nice feature, and one of the prominent features of Bootstrap sites. At the core, the navbar includes styling for site names, and basic navigation. It can later be extended by adding form specific controls, and specialized dropdowns. To be sure that the navbar is constrained to the width of the content of the page, either place it inside of a `.span12` or the `.container` class.



Figure 3-21. Basic Navbar Example

Basic Navbar Code Example.

```
<div class="navbar">
  <div class="navbar-inner">
    <a class="brand" href="#">Title</a>
    <ul class="nav">
      <li class="active"><a href="#">Home</a></li>
      <li><a href="#">Link</a></li>
      <li><a href="#">Link</a></li>
    </ul>
  </div>
</div>
```

In the code above, note the `.brand` class, this will give the text a lighter font-weight and slightly larger size.

Brand Class Example.

```
<a class="brand" href="#">Project name</a>
```

Nav Links

To add links to the navbar, simply add an unordered list with the class of `.nav`. If you want to add a divider to your links, you can do that by adding an empty list-item with a class of `.divider-vertical`.



Figure 3-22. Nav Links

Navbar Links Code Example.

```
<ul class="nav">
  <li class="active"><a href="#">Home</a></li>
  <li><a href="#">First Link</a></li>
  <li><a href="#">Second Link</a></li>
  <li class="divider-vertical"></li>
  <li><a href="#">Third Link</a></li>
</ul>
```

Forms

Instead of using the default, class based forms from chapter 2, forms that are in the navbar use the `.navbar-form` class. This ensures that the forms margins are properly set, and match the nav stylings. Of note, `.pull-left`, and `.pull-right` helper classes may help move the form in the proper position.



Figure 3-23. Default Navbar Form

Default Navbar Form Styling.

```
<form class="navbar-form pull-left">
  <input type="text" class="span2" id="fname">
  <button type="submit" class="btn">
</form>
```

To add rounded corners, taking style cues from the search inputs of iOS devices, instead of using `.navbar-form`, use the `.navbar-search` class.



Figure 3-24. Navbar Search Input

Navbar Search Input Code Example.

```
<form class="navbar-search" accept-charset="utf-8">
  <input type="text" class="search-query" placeholder="Search">
</form>
```

Navbar Menu Variations

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 to the top or bottom of the page, or have it scroll statically with the page.

Fixed to the top

If you want the navbar fixed to the top, simply add `.navbar-fixed-top` to the `.navbar` class. To prevent the navbar from sitting on top of other content in the body of the page, add at least 40 pixels of padding to the `<body>` tag.

Fixed Top Navbar.

```
<div class="navbar navbar-fixed-top">
  <div class="navbar-inner">
    <a class="brand" href="#">Title</a>
    <ul class="nav">
      <li class="active"><a href="#">Home</a></li>
      <li><a href="#">Link</a></li>
      <li><a href="#">Link</a></li>
    </ul>
  </div>
</div>
```

Fixed Bottom Navbar

To affix the navbar to the bottom of the page, simply add `.fixed-navbar-bottom` class to the navbar. Once again, to prevent overlap, add at least 40 pixels of padding to the `<body>` tag.

Fixed Bottom Navbar.

```
<div class="navbar navbar-fixed-bottom">
  <div class="navbar-inner">
    <a class="brand" href="#">Title</a>
    <ul class="nav">
      <li class="active"><a href="#">Home</a></li>
      <li><a href="#">Link</a></li>
      <li><a href="#">Link</a></li>
    </ul>
  </div>
</div>
```

Static Top Navbar

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.

Static Top Navbar.

```
<div class="navbar navbar-static-top">
  <div class="navbar-inner">
    <a class="brand" href="#">Title</a>
    <ul class="nav">
      <li class="active"><a href="#">Home</a></li>
      <li><a href="#">Link</a></li>
      <li><a href="#">Link</a></li>
    </ul>
  </div>
</div>
```

Responsive Navbar

Like the rest of Bootstrap, the navbar can be totally responsive. To add the responsive features, the content that you want to be collapsed needs to be wrapped in a `<div>` with `.nav-collapse.collapse` as a class. The collapsing nature is tripped by a button that has a the class of `.btn-navbar` 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` tells which element to toggle. In the example below, three `` with a class of `.icon-bar` create what I like to call the hamburger button. This will toggle the elements that in the `.nav-collapse <div>`. For this to work, the `bootstrap-responsive.css`, and either the `collapse.js` or the full `bootstrap.js` files must be included for this feature to work.



Figure 3-25. Responsive Navbar

Responsive Navbar Code Example.

```
<div class="header">
  <div class="navbar-inner">
    <div class="container">
      <a class="btn btn-navbar" data-toggle="collapse" data-target=".nav-collapse">
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </a>
      <!-- Leave the brand out if you want it to be shown when other elements are collapsed -->
      <a href="#" class="brand">Project Name</a>
    </div>
  </div>
</div>
```

```

        <!-- Everything that you want collapsed, add it to the collapse div. -->
        <div class="nav-collapse collapse">
            <!-- .nav, .navbar-search etc... -->
        </div>
    </div>
</div>

```

Inverted Navbar

To create an inverted navbar, where the background is black, with white text, simply add `.navbar-inverse` to the `.navbar` class.

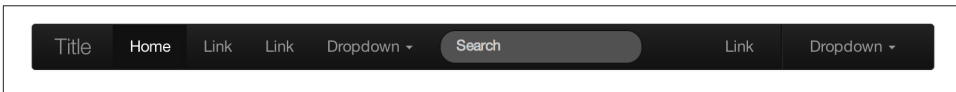


Figure 3-26. Inverted Navbar

Inverted Navbar Code Example.

```

<div class="navbar navbar-inverse">
    ...
</div>

```

Breadcrumbs

Breadcrumbs are a great way to show hierarchy based information for a site. In the case of blogs, it could show the dates of publishing, categories or tags, or for a full CMS, any type of information. A breadcrumb in Bootstrap is simply an unordered list with a class of `.breadcrumb`. There is also a helper class of `.divider` that mutes the colors and makes the text a little smaller too. You could use forward slashes, arrows, or any divider that you choose. Note that the divider here in the breadcrumbs has slightly different markup than the navbar example.



Figure 3-27. Breadcrumb Example

Breadcrumb Code Example.

```
<ul class="breadcrumb">
  <li><a href="#">Home</a> <span class="divider"></span></li>
  <li><a href="#">2012</a> <span class="divider"></span></li>
  <li><a href="#">December</a> <span class="divider"></span></li>
  <li><a href="#">5</a></li>
</ul>

<ul class="breadcrumb">
  <li><a href="#">Home</a> <span class="divider">&rarr;</span></li>
  <li><a href="#">Dinner Menu</a> <span class="divider">&rarr;</span></li>
  <li><a href="#">Specials</a> <span class="divider">&rarr;</span></li>
  <li><a href="#">Steaks</a></li>
</ul>

<ul class="breadcrumb">
  <li><a href="#">Home</a> <span class="divider">&raquo;</span></li>
  <li><a href="#">Electronics</a> <span class="divider">&raquo;</span></li>
  <li><a href="#">Raspberry Pi</a></li>
</ul>
```

Pagination

Bootstrap handles pagination like a lot of interface elements, an unordered list, with wrapper `<div>` that has a specific class that identifies the element. In the basic form, adding `.pagination` do the parent `<div>` creates a row of bordered links. Each of the list items can be additionally styled by using the `.disabled` or `.active` class.



Figure 3-28. Basic Pagination Example

Basic Pagination Code Examp.

```
<div class="pagination">
  <ul>
    <li><a href="#">Prev</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="#">Next</a></li>
  </ul>
</div>
```



Figure 3-29. *Pagination with helper classes*

Pagination with helper classes code examples.

```

<div class="pagination pagination-centered">
  <ul>
    <li class="disabled"><a href="#"></a></li>
    <li class="active"><a href="#">1</a></li>
    <li><a href="#">2</a></li>
    <li><a href="#">3</a></li>
    <li><a href="#">4</a></li>
    <li><a href="#">5</a></li>
    <li><a href="#">></a></li>
  </ul>
</div>

```

In addition to the `.active` and `.disabled` classes for list items, you can also add `.pagination-centered` to the parent `<div>`. This will center the contents of the `div`. If you want the items right aligned in the `<div>` add `.pagination-right`. For sizing, in addition to the normal size, there are three other sizes, applied by adding a class to the wrapper `<div>`. They are: `.pagination-large`, `pagination-small` and `pagination-mini`.



Figure 3-30. *Pagination Sizes*

Pagination Code Example.

```

<div class="pagination pagination-large">
  <ul>
    ...
  </ul>
</div>
<div class="pagination">
  <ul>
    ...

```

```

    </ul>
  </div>
  <div class="pagination pagination-small">
    <ul>
      ...
    </ul>
  </div>
  <div class="pagination pagination-mini">
    <ul>
      ...
    </ul>
  </div>

```

Pager

If you need to create simple pagination links that go beyond text, the pager can work quite well. Like the pagination links, the markup is an unordered list that sheds the wrapper `<div>`. By default, the links are centered.

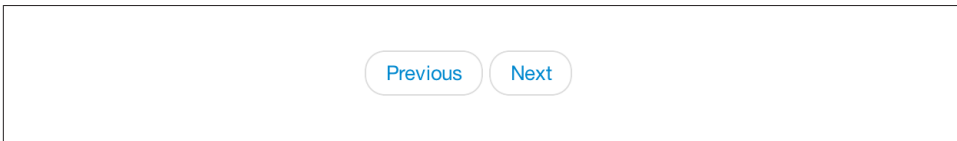


Figure 3-31. Basic Pager

Basic Pager Code Example.

```

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

```

To left/right align the different links, you just need to add the `.previous` and `.next` class to the list-items. Also, like `.pagination` above, you can add the `disabled` class for a muted look.



Figure 3-32. Aligned Page Links

Aligned Page Links Code Example.

```

<ul class="pager">
  <li class="previous">

```



```

    <a href="#">&larr; Older</a>
  </li>
  <li class="next">
    <a href="#">Newer &rarr;</a>
  </li>
</ul>

```

Labels and Badges

Labels and Badges are great for offering counts, tips, or other markup for pages. Another one of my favorite little Bootstrap touches.



Figure 3-33. Labels

Label Markup.

```

<span class="label">Default</span>
<span class="label label-success">Success</span>
<span class="label label-warning">Warning</span>
<span class="label label-important">Important</span>
<span class="label label-info">Info</span>
<span class="label label-inverse">Inverse</span>

```

Badges

Badges are similar to labels, the primary difference is that they have more rounded corners. The colors of badges reflect the same classes as labels.



Figure 3-34. Badges

Badges Code Example.

```

<span class="badge">1</span>
<span class="badge badge-success">2</span>
<span class="badge badge-warning">4</span>
<span class="badge badge-important">6</span>
<span class="badge badge-info">8</span>
<span class="badge badge-inverse">10</span>

```

Typographic Elements

In addition to buttons, labels, forms, tables and tabs, Bootstrap has a few more elements for basic page layout. The hero unit is a large, content area that increased the size of headings, and adds a lot of margin for landing page content. To use, simply create a container `<div>` with the class of `.hero-unit`. In addition to a larger `<h1>`, all the font-weight is reduced to 200.

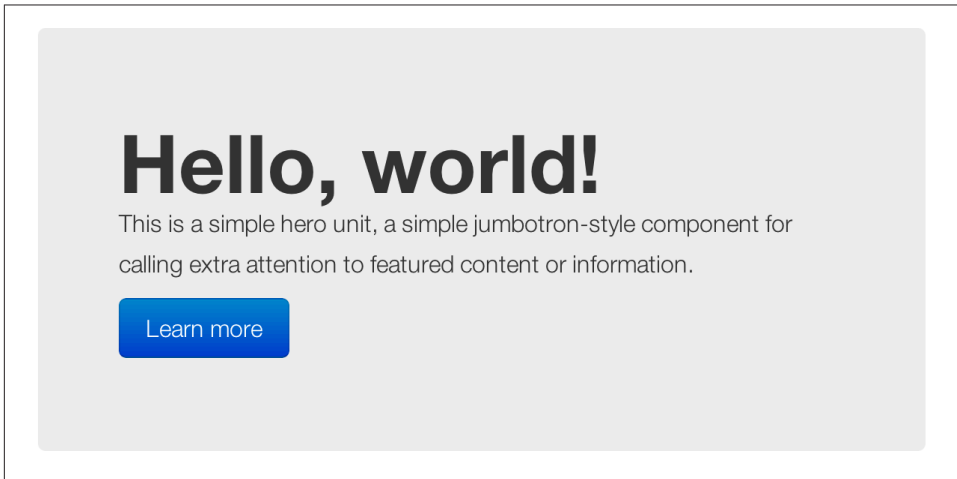


Figure 3-35. Hero Unit

Hero Unit Code Example.

```
<div class="hero-unit">
  <h1>Heading</h1>
  <p>Tagline</p>
  <p>
    <a class="btn btn-primary btn-large">Learn more</a>
  </p>
</div>
```

Page Header

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

Example page header Subtext for header

Figure 3-36. Page Header

Page Header Code Example.

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

Thumbnails

A lot of sites need a way to layout images in a grid, and Bootstrap has an easy way to do this. At the simplest, you add an `<a>` tag with the class of `.thumbnail` around an image. This adds four pixels of padding, and a grey border. On hover, an animated glow is added around the image.



Figure 3-37. Basic Thumbnail

Thumbnail Code Example.

```
<a href="#" class="thumbnail">  
    
</a>
```

To add more content to the markup, as an example, you could add headings, buttons and more, swap the `<a>` tag that has a class of `.thumbnail` to be a `<div>`. Inside of that `<div>`, you can add anything you need. Since 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 left.

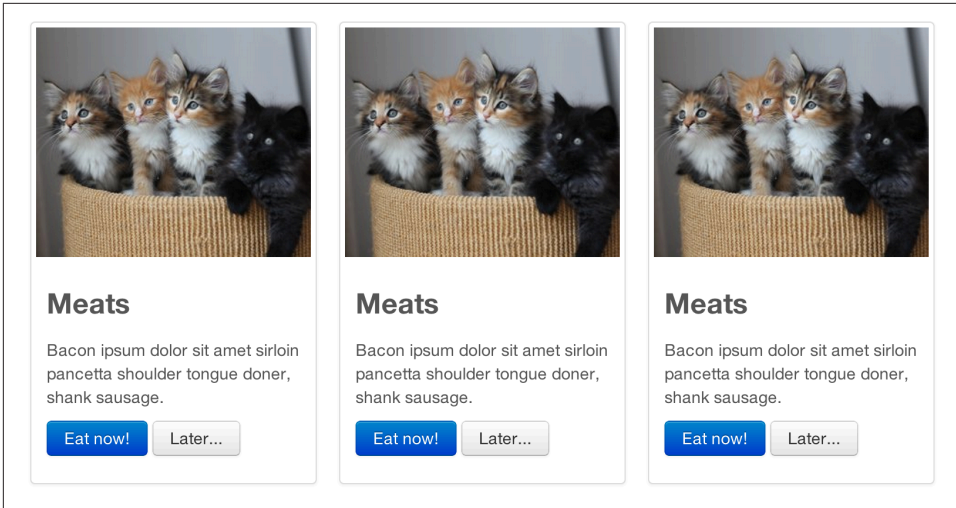


Figure 3-38. Extended Thumbnail

Customizable Code Example.

```

<ul class="thumbnails">
  <li class="span4">
    <div class="thumbnail">
      
      <div class="caption">
        <h3>Meats</h3>
        <p>Bacon ipsum dolor sit amet sirloin pancetta shoulder tongue doner, shank sausage.</p>
        <p><a href="#" class="btn btn-primary">Eat now!</a> <a href="#" class="btn">Later...</a></p>
      </div>
    </div>
  </li>
  <li class="span4">
    ...
  </li>
</ul>

```

Alerts

Like the modals described in the next chapter, alerts provide a way to style messages to the user. The default alert is created by creating a wrapper `<div>` and adding a class of `.alert`.

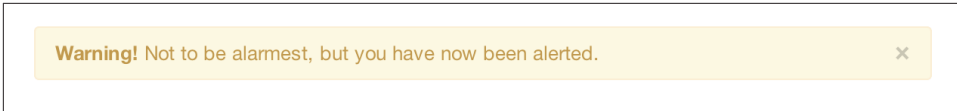


Figure 3-39. Basic Alert

Basic Alert Code Example.

```
<div class="alert">  
  <a href="#" class="close" data-dismiss="alert">&times;</a>  
  <strong>Warning!</strong> Not to be alarmist, but you have now been alerted.  
</div>
```

The alert uses the alerts jquery plugin that is covered in chapter 4. To close the alert, you can use a button that contains the `data-dismiss="alert"` attribute. Mobile Safari, and Mobile Opera browsers require an `href="#"` to close.

If you have a longer message in your alert, you can use the `.alert-block` class. This provides a little more padding above and below the content contained in the alert, particularly useful for multi page lines of content.

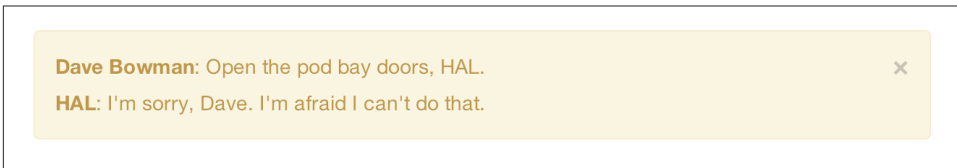


Figure 3-40. Alert Block

There are also three other color options, to help provide a more semantic method for the alert. They are added by adding either `.alert-error`, `.alert-success`, or `alert-info`.

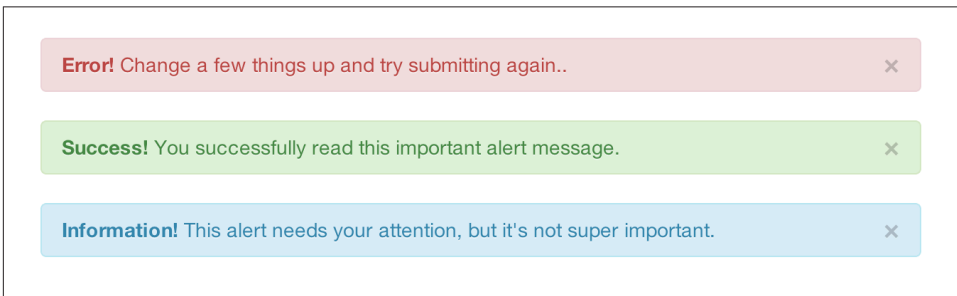


Figure 3-41. Alert Color Options

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. Personally, I think that these elements are more an exercise in markup, and have little purpose beyond that in the Bootstrap framework. That being said, with thousand of people using Bootstrap, there are likely a few outliers that have a good reason. By nature, these are static elements, and need some sort of Javascript interaction to provide any interaction.

The default progress bar has a light background and a blue progress bar. To create, add a `<div>` with a class of `.progress`. And then inside, add an empty `<div>` with a class of `.bar`. Add a style attribute with the width in percentage. In the case below, I added `style="width: 60%;"`; to indicate that the progress bar was at 60%.

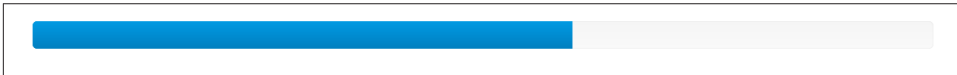


Figure 3-42. Default Progress Bar

Progress Bar Example.

```
<div class="progress">  
  <div class="bar" style="width: 60%;"></div>  
</div>
```

To create a striped progress bar, just add `.progress-striped` to the container `<div>`. Of note, striped progress bars are not available in Internet Explorer 7 and 8.



Figure 3-43. Striped Progress Bar

Striped Progress Bar Code Example.

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

Like the striped version of the progress bar, you can animate the stripes, making them look like the blue light special barbershop pole.

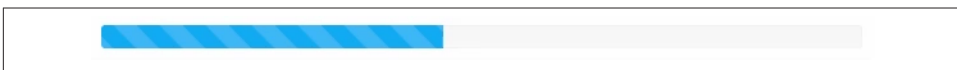


Figure 3-44. Animated Progress Bar

Animated Progress Bar Code Example.

```
<div class="progress progress-striped active">
  <div class="bar" style="width: 40%;"></div>
</div>
```

In addition to the blue progress bar, there are options for green, yellow, and red by using the `.bar-success`, `bar-warning`, and `bar-danger` classes. Progress bars can be stacked, indicating a graph of sorts by adding multiple elements together like so:



Figure 3-45. Stacked Progress Bar

Stacked Progress Bar Example.

```
<div class="progress">
  <div class="bar bar-success" style="width: 35%;"></div>
  <div class="bar bar-warning" style="width: 20%;"></div>
  <div class="bar bar-danger" style="width: 10%;"></div>
</div>
```

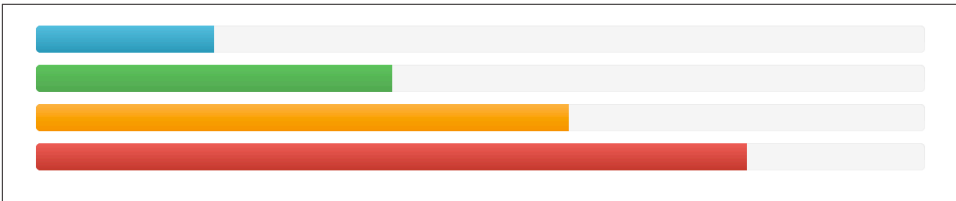


Figure 3-46. Progress Bar Color Examples

Media Object

When you look at social sites like Facebook, Twitter and others, strip away some of the formatting from timelines, and you will see the Media Object. Driven by the Bootstrap community, and based on principles from the oocss [<http://oocss.org/>] community, the goal of the media element is to make the code for developing these blocks of information drastically shorter. Nicole Sullivan-Hass shares a few elements of the media object like this on her site stubbornella.org. The media object is designed to literally save hundreds of lines of code, making it easy to customize.



Figure 3-47. Media Object Examples

Bootstrap leaves the design and formatting to you, but provides a simple way to get going. Like a lot of other tools in Bootstrap, the markup is light, and the end goal is reached by applying classes to some simple markup. There are two forms to the media object, the `.media`, and the `.media-list`. If you are preparing a list where the items would be part of an unordered list, use the `.media-list`. Using just `<div>` elements, use the `.media` object.

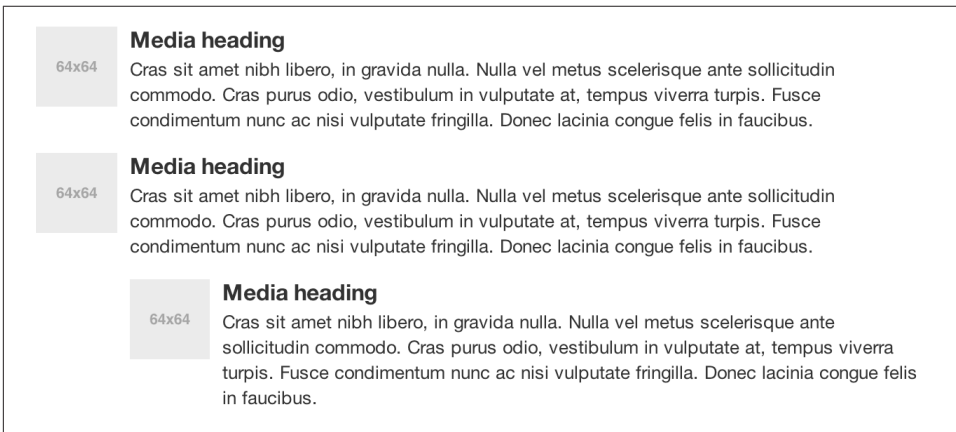


Figure 3-48. Default Media Object

Media Object Code Example.

```

<div class="media">
  <a class="pull-left" href="#">
    
  </a>
  <div class="media-body">

```



```

<h4 class="media-heading">Media heading</h4>
<p>...</p>

<!-- Nested media object -->
<div class="media">
  ...
</div>
</div>
</div>

```

To use the media list, change the container `<div>` to an `` and add the class `.media-list`. Since you can nest media objects, it makes it handy markup for comments, or other lists.

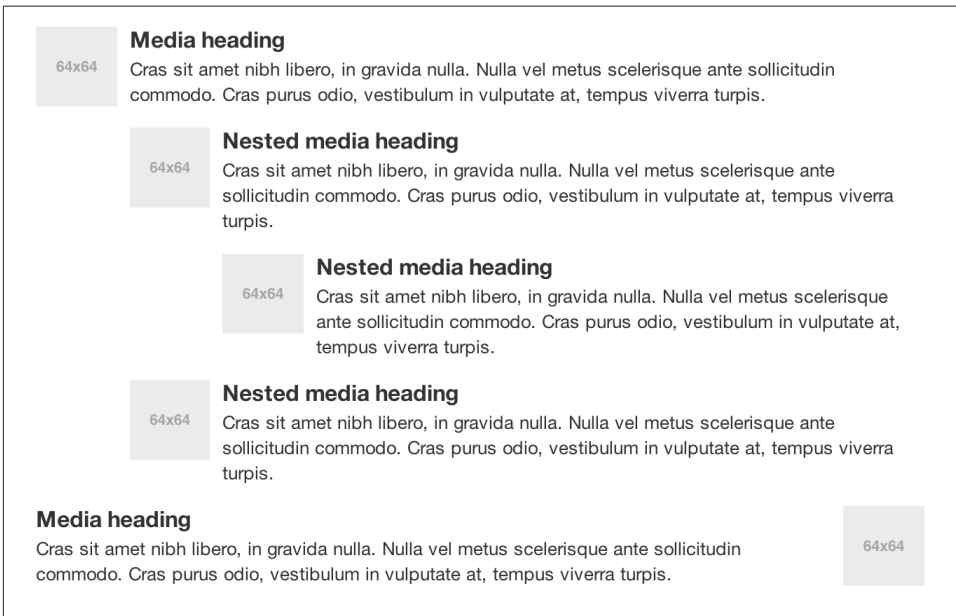


Figure 3-49. Media List Example

Media List Code Example.

```

<ul class="media-list">
  <li class="media">
    <a class="pull-left" href="#">
      
    </a>
    <div class="media-body">
      <h4 class="media-heading">Media heading</h4>
      <p>...</p>
      ...
    </div>
  </li>
  ...
</ul>

```

```
<!-- Nested media object -->
<div class="media">
  ...
</div>
</div>
</li>
</ul>
```

Misc

Now, at the end of chapter 3, there are a few more components that Bootstrap offers. There are a few that are layout based, and a few that layout based, and a few that are production based helper classes. The first among these are the wells.

A well is a container `<div>` that add some styles to appear sunken on the page. I have used them before for blog post meta information, like author, date, categories. To create, simply wrap the content that you would like to appear in the well with a `<div>` containing the class of `.well`.



Figure 3-50. Well

Well Example.

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

There are two additional classes that can be used in conjunction with `.well`, `.well-large` and `.well-small`. These affect the padding, making the well larger or smaller depending on the class.



Figure 3-51. Well Optional Classes

```
<div class="well well-large">
    Look, I'm in a .well-large!
</div>

<div class="well well-small">
    Look, I'm in a .well-small!
</div>
```

Helper Classes

Pull Left

To float an element to the left, use the `.pull-left` class.

Pull Left HTML.

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

Pull Left CSS.

```
.pull-left {
    float: left;
}
```

Pull Right

To float an element to the right, use the `.pull-right` class.

Pull Right CSS.

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

Pull Right CSS.

```
.pull-right {
    float: right;
}
```

Clearfix

To clear the float of any element, use the `.clearfix` class. When you have to elements that are floated alongside each other that are different sizes, it is necessary to force the following elements below, or to *clear* the preceding content. You can do this with a simple empty `<div>` the class of `.clearfix`.

Clearfix HTML.

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

Clearfix CSS.

```
.clearfix {  
  *zoom: 1;  
  &:before,  
  &:after {  
    display: table;  
    content: "";  
  }  
  &:after {  
    clear: both;  
  }  
}
```

Bootstrap Javascript Plugins

Most of the components discussed in the previous chapter are just the beginning. Bootstrap comes bundled with 13 jQuery plugins that extend the features, and can add more interaction to your site. To get started with the Bootstrap Javascript plugins, you don't need to be an advanced Javascript developer. In fact, most of the plugins can be enacted without writing a single line of code by utilizing the Bootstrap data API.

Overview

Bootstrap can be included on your site in two forms, either compiled or raw. Bootstrap 2.2.2 the uncompressed file is 59kb, and the minimized version is only 32kb. All of the Bootstrap plugins are accessible entirely using the included data API, with this, you don't need to include a single line of javascript to invoke any of the plugins features.

Typically, Javascript lies in either a separate file, or at the bottom of the page before the closing `</body>` tag. You can either use the `src` attribute to link to another file, or write the contents of the file between the opening and closing tags.

Javascript Usage.

```
<!-- To reference another Javascript file -->
<script src="assets/js/javascript.js"></script>

<!-- To write Javascript to the page -->
<script type="text/javascript">
    function js_alert{
        alert('Page has loaded');
    }
</script>
```

Generally, it is best to include all Javascript calls into check that ensures that the DOM has been loaded on the page. As the browser parses the page, if you have the Javascript

trying to fire earlier, it may miss elements. With jQuery, this is easily done by selecting the document, or the entire content of the page, and then applying the `.ready()` method.

jQuery Ready Method.

```
$(document).ready(function(){
    alert('Page has loaded');
    // Once the page has loaded, and is ready, an alert will fire.
});
```

Like I mentioned above, Bootstrap has a data API where you can write data attributes into the HTML of the page. If you need to turn off the data API, you can unbind the attributes by adding this line of Javascript.

Disable Bootstrap Javascript Data API.

```
$('#body').off('.data-api')
```

If you need to disable a single plugin, you can do it programmatically using the namespace of the plugin along with the `data-api` namespace.

Disable an Individual Plugin.

```
$('#body').off('.alert.data-api')
```

Programmatic API

The developers of Bootstrap believe that you should be able to use all of the plugins entirely through the Javascript API. All public APIs are single, chainable methods, and return the collection acted upon.

```
$('.btn.danger').button('toggle').addClass('active')
```

All methods should accept an optional options object, a string which targets a particular method, or nothing (which initiates a plugin with default behavior).

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

Transitions

The transition plugin provides simple transition effects. A few examples include:

- Sliding or fading in modals
- Fading out tabs
- Fading out alerts
- Sliding carousel panes

Modal

A modal is a child window that layered over its parent window. Typically, the point is to display content from a separate source that can have some interaction without leaving the parent window. They can provide information, interaction, or more for a page.. I have used that as a window for holding slideshows, login/registration information, and more. The Modal plugins is probably one of my favorite Bootstrap features.

To create a static example, use the code below.

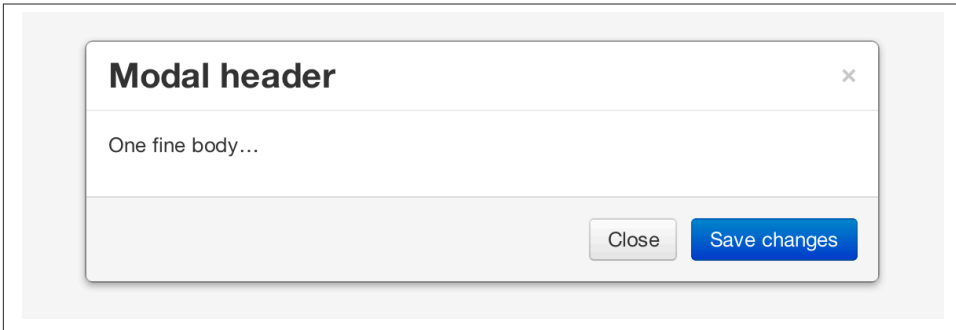


Figure 4-1. Static Modal Window

Static Modal Window Code Example.

```
<div class="modal hide fade">
  <div class="modal-header">
    <button type="button" class="close" data-dismiss="modal" aria-hidden="true">&times
    <h3>Modal header</h3>
  </div>
  <div class="modal-body">
    <p>One fine body...</p>
  </div>
  <div class="modal-footer">
    <a href="#" class="btn">Close</a>
    <a href="#" class="btn btn-primary">Save changes</a>
  </div>
</div>
```

To invoke the modal window, you need to have some kind of a trigger. Normally I use a button or an link. If you look in the code below, you will see that the `<a>` tag, the `href="myModal"` is the target of the modal that you want to load on the page. The 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 can create many on the page to be loaded at different times.

There are three classes to take note of in the modal. The first is `modal`. This should make sense, but this is identifying the content of the `div` as a modal. The second is `hide`. This tells the browser to hide the content of the `div` until we are ready to invoke it. And last, the `fade` class. When the modal is toggled, it will cause the content to fade in and out.

```
<!-- Button to trigger modal -->
<a href="#myModal" role="button" class="btn" data-toggle="modal">Launch demo modal</a>

<!-- Modal -->
<div id="myModal" class="modal hide fade" tabindex="-1" role="dialog" aria-labelledby="myModalLabel"
  <div class="modal-header">
    <button type="button" class="close" data-dismiss="modal" aria-hidden="true">×</button>
    <h3 id="myModalLabel">Modal header</h3>
  </div>
  <div class="modal-body">
    <p>One fine body...</p>
  </div>
  <div class="modal-footer">
    <button class="btn" data-dismiss="modal" aria-hidden="true">Close</button>
    <button class="btn btn-primary">Save changes</button>
  </div>
</div>
```

Using Modal through Data Attributes

Using the Bootstrap Javascript Data API, you simply need to pass a few data attributes to toggle the slideshow. To start with, set `data-toggle="modal"` on the link or button that you want to use to invoke the modal and then `data-target="#foo"` to the ID of the modal that you'd like to use.

Using Modal through Javascript

To call a modal with `id="myModal"` use a single line of JavaScript:

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

Modal Options

Options can either be passed in via data attributes, or with Javascript. To use the data attributes, prepend `data-` to the option name, like `data-backdrop=""`.

Table 4-1. Modal Options

Name	Type	Default	Description
backdrop	boolean	true	Set to false if you don't want the modal to be closed when they click outside of the modal.
keyboard	boolean	true	Closes the modal when escape key is pressed, set to false to disable.
show	boolean	true	Shows the modal when initialized.
remote	path	false	Using the jQuery .load method, inject content into the modal body. If an href with a valid URL is added, that will load that content.

Methods

Options

Activates your content as a modal. Accepts an optional options object.

`.modal(options).`

```
$('#myModal').modal({  
    keyboard: false  
})
```

Toggle

Manually toggles a modal.

`.modal(toggle).`

```
$('#myModal').modal('toggle')
```

Show

Manually opens a modal.

`.modal(show).`

```
$('#myModal').modal('show')
```

Hide

Manually hides a modal.

`.modal(hide).`

```
$('#myModal').modal('hide')
```

Events

If you need specific events during the firing events of Bootstrap's modals, you can use the following events.

Table 4-2. Modal Events

Event	Description
show	Fired after the show method is called.
shown	Fired when the modal has been made visible to the user.
hide	Fired when the hide instance method has been called.
hidden	Fired when the modal has finished being hidden from the user.

```
$('#myModal').on('hidden', function () {  
    alert('Hey girl, I heard you like modals...');  
})
```

Dropdown

The dropdown was covered extensively in chapter 3, but then, the interaction was simply glossed over. As a refresher, dropdowns can be added to the the navbar, pills, tabs and buttons.

Dropdown Usage via the Data API

To use, add use `data-toggle="dropdown"` to a link or button to toggle the dropdown.

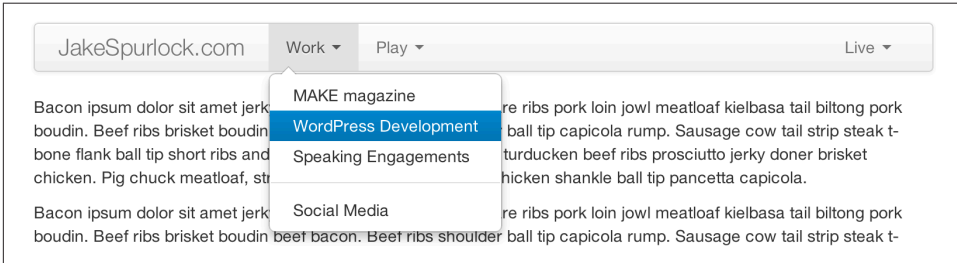


Figure 4-2. Dropdown Within Navbar

Dropdown Code Example with Data Attributes.

```
<li class="dropdown">
  <a href="#" id="drop" role="button" class="dropdown-toggle" data-toggle="dropdown">Word <b>
  <ul class="dropdown-menu" role="menu" aria-labelledby="drop">
    <li><a tabindex="-1" href="#">MAKE magazine</a></li>
    <li><a tabindex="-1" href="#">WordPress Development</a></li>
    <li><a tabindex="-1" href="#">Speaking Engagements</a></li>
    <li class="divider"></li>
    <li><a tabindex="-1" href="#">Social Media</a></li>
  </ul>
</li>
```

If you need to keep links intact, useful if the browser is not enabling Javascript, use the `data-target` attribute along with `href="#"`.

Dropdown via the `data-target` Attribute.

```
<div class="dropdown">
  <a class="dropdown-toggle" id="dLabel" role="button" data-toggle="dropdown" data-target="#">
    Dropdown
    <b class="caret"></b>
  </a>
  <ul class="dropdown-menu" role="menu" aria-labelledby="dLabel">
    ...
  </ul>
</div>
```

Dropdown Usage via Javascript

To call the dropdown toggle via Javascript, use the following method.

Dropdown via Javascript.

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

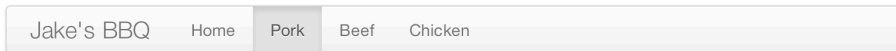
Methods

The dropdown toggle has a simple method to show or hide the dropdown. There are no options.

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

Scrollspy

The scrollspy plugin allows you to target sections of the page based on scroll position. In its basic implementation, as you scroll you can add active classes to the nav bar based on the scroll position. To add the scrollspy plugin via data attributes, add `data-spy="scroll"` to the element you want to spy on (most typically this would be the body) and `data-target=".navbar"` to the navbar that you want to apply the class changes to. 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.



Bacon ipsum dolor sit amet jerky flank andouille, ham hock spare ribs pork loin jowl meatloaf kielbasa tail biltong pork boudin. Beef ribs brisket boudin beef bacon. Beef ribs shoulder ball tip capicola rump. Sausage cow tail strip steak t-

Usage

Body Configuration for Scrollspy.

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

In the navbar, you will need to have page anchors that will serve as indicators for the element to spy on.

Navbar Setup for Scrollspy.

```
<div class="navbar">
  <div class="navbar-inner">
    <div class="container">
      <a class="brand" href="#">Jake's BBQ</a>
      <div class="nav-collapse">
        <ul class="nav">
          <li class="active"><a href="#">Home</a></li>
          <li><a href="#pork">Pork</a></li>
          <li><a href="#beef">Beef</a></li>
```

```

        <li><a href="#chicken">Chicken</a></li>
    </ul>
</div><!-- /.nav-collapse -->
</div>
</div><!-- /navbar-inner -->
</div>

```

Usage via Javascript

Javascript Usage.

```
$('#navbar').scrollspy()
```

Scrollspy Methods

.scrollspy(*refresh*)

When calling the scrollspy via the Javascript method, you will need to call the .refresh method to update the DOM. This is helpful if any elements of the DOM have changed.

```

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

```

Options

Options can be passed via data attributes or JavaScript. For data attributes, append the option name to data-, as in data-offset="".

Table 4-3. Scrollspy Options

Name	type	default	description
offset	number	10	Pixels to offset from top of page when calculating position of scroll.

The offset option is handy when you are using a fixed navbar. You will want to offset the scroll by about 50 pixels so that it reads at the correct time.

Events

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

Toggleable Tabs

That tabbable tabs were introduced back in chapter 3. Combing a few data attributes, you can easily create a tabbed interface. To do so, create the nav interface, and then wrap the content of the tabs inside a <div> with a class of .tab-content.

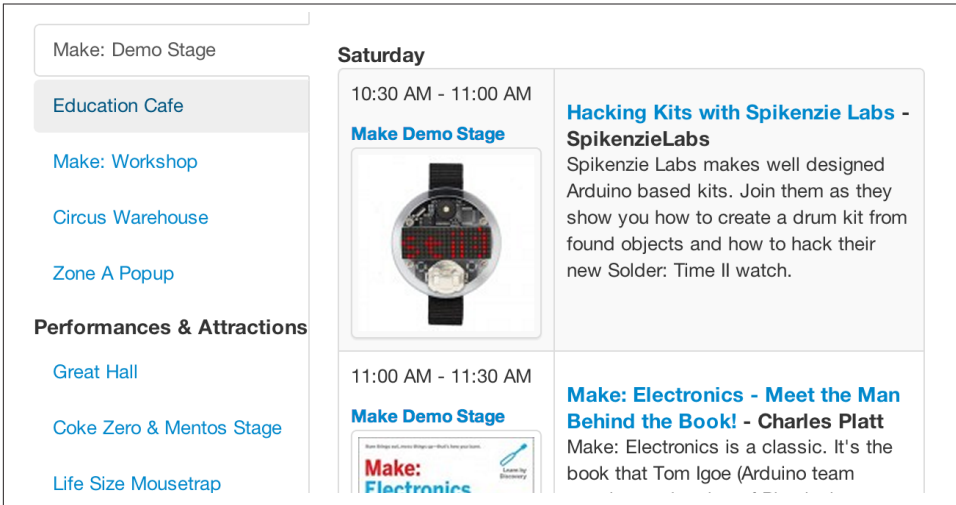


Figure 4-3. Toggleable Tabs

Basic Markup of Toggleable Tabs.

```

<ul class="nav nav-tabs">
  <li><a href="#home" data-toggle="tab">Home</a></li>
  <li><a href="#profile" data-toggle="tab">Profile</a></li>
  <li><a href="#messages" data-toggle="tab">Messages</a></li>
  <li><a href="#settings" data-toggle="tab">Settings</a></li>
</ul>

<div class="tab-content">
  <div class="tab-pane active" id="home">...</div>
  <div class="tab-pane" id="profile">...</div>
  <div class="tab-pane" id="messages">...</div>
  <div class="tab-pane" id="settings">...</div>
</div>

```

Usage

To enable the tabs, you can use the Bootstrap Data API, or using Javascript directly. With the Data API, you need to add `data-toggle` to the anchors. The anchor targets will activate the the element that has the `.tab-pane` class and relative ID. Alternatively, `data-target=""` may be used instead of `href="#"` to apply the same action.

Enable Tabs via Javascript.

```

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

```

Example of different ways to activate tabs.

```
$('#myTab a[href="#profile"]').tab('show'); // Select tab by name
$('#myTab a:first').tab('show'); // Select first tab
$('#myTab a:last').tab('show'); // Select last tab
$('#myTab li:eq(2) a').tab('show'); // Select third tab (0-indexed)
```

Events

Event	Description
show	This event fires on tab show, but before the new tab has been shown. Use <code>event.target</code> and <code>event.relatedTarget</code> to target the active tab and the previous active tab (if available) respectively.
shown	This event fires on tab show after a tab has been shown. Use <code>event.target</code> and <code>event.relatedTarget</code> to target the active tab and the previous active tab (if available) respectively.

Example of shown method.

```
$('#a[data-toggle="tab"]').on('shown', function (e) {
    e.target // activated tab
    e.relatedTarget // previous tab
})
```

For more information about the jQuery `.on` method, read more at the jQuery website. [jQuery.on](#)

Tooltips

Tooltips are useful when you need to describe a link, or perhaps used in conjunction with the `<abbr>` tag, provide the definition of an abbreviation. The plugin was originally based on the jQuery.tipsy plugin written by Jason Frame. It has since been updated to work without images, animate with a CSS animation, and work with the Bootstrap Javascript API.

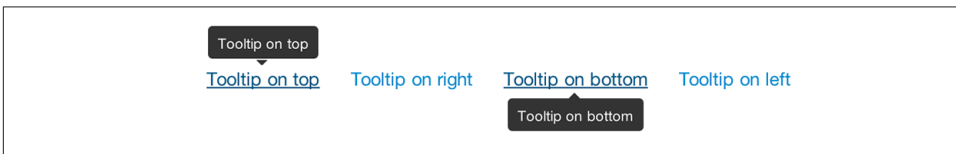


Figure 4-4. Tooltip Placement

Usage

Bootstrap Data API.

```
<a href="#" rel="tooltip" title="This is the tooltip">Tooltip Example</a>
```

Javascript.

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

Options

Like all of the plugins, there are options that can be added via the Bootstrap Data API, or invoked via Javascript. All options need to have `data-` appended to them. So, the `title` option would become `data-title`.

Name	type	default	description
<code>animation</code>	boolean	<code>true</code>	apply a css fade transition to the tooltip
<code>html</code>	boolean	<code>false</code>	Insert html into the tooltip. If false, jquery's <code>text</code> method will be used to insert content into the dom. Use text if you're worried about XSS attacks.
<code>placement</code>	string/function	<code>'top'</code>	how to position the tooltip - top/bottom/left/right
<code>selector</code>	string	<code>false</code>	If a selector is provided, tooltip objects will be delegated to the specified targets.
<code>title</code>	string/function	<code>''</code>	default title value if <code>title</code> tag isn't present
<code>trigger</code>	string	<code>'hover'</code>	how tooltip is triggered - click
<code>hover</code>	<code>focus</code>	<code>manual</code>	<code>delay</code>

Methods

Options

Attaches a tooltip handler to an element collection.

```
$('.element').tooltip(options)
```

Show

Reveals an element's tooltip.

```
$('#element').tooltip('show')
```

Hide

Hides an element's tooltip.

```
$('#element').tooltip('hide')
```

Toggle

Toggles an element's tooltip.

```
$('#element').tooltip('toggle')
```

Destroy

Hides and destroys an element's tooltip.

```
$('#element').tooltip('destroy')
```

Popover

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

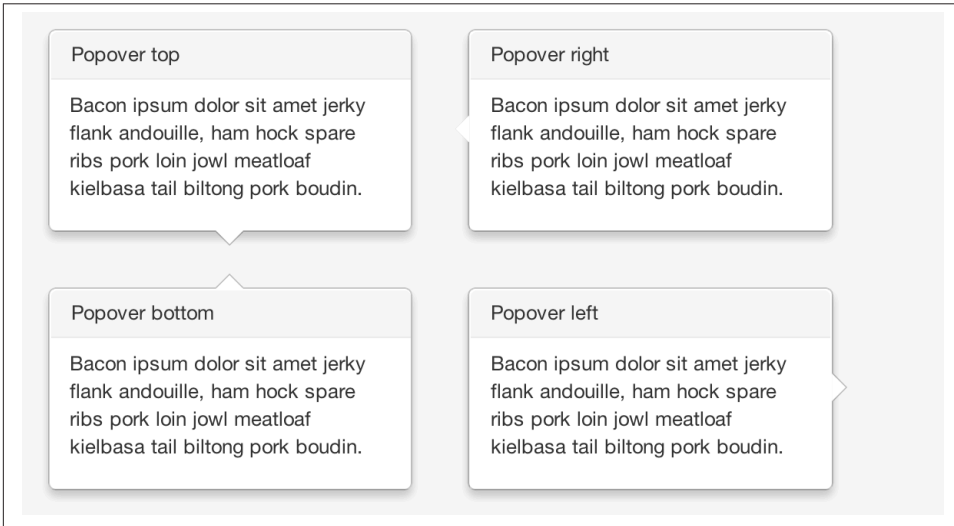


Figure 4-5. Popover Placement

```
<a href="#" class="btn" rel="popover" title="Using Popover" data-content="Just add content to the
```

Usage

Enable with Javascript.

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

Options

All options can be passed via the Bootstrap Data API, or directly with Javascript.

Name	type	default	description
animation	boolean	true	apply a css fade transition to the tooltip
html	boolean	false	Insert html into the popover. If false, jquery's <code>text</code> method will be used to insert content into the dom. Use <code>text</code> if you're worried about XSS attacks.
placement	string function	'right'	how to position the popover - top bottom left right

Name	type	default	description
selector	string	false	if a selector is provided, tooltip objects will be delegated to the specified targets
trigger	string	'click'	how popover is triggered - click hover focus manual
title	string function	"	default title value if `title` attribute isn't present
content	string function	"	default content value if `data-content` attribute isn't present
delay	number object	0	delay showing and hiding the popover (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 }

Methods

Options

Initializes popovers for an element collection.

```
$().popover(options)
```

Show

Reveals an elements popover.

```
$(">#element).popover(">'show')
```

Hide

Hides an elements popover.

```
$(">#element).popover(">'hide')
```

Toggle

Toggles an elements popover.

```
$(">#element).popover(">'toggle')
```

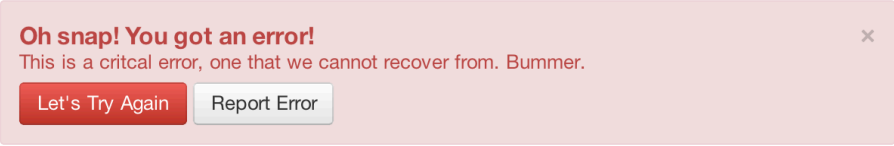
Destroy

Hides and destroys an element's popover.

```
$(">#element).popover(">'destroy')
```

Alerts

With the Data API, it is easy to add dismiss functionality to alert messages.



Usage

Dismiss Via Javascript.

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

Dismiss Via Data API.

```
<a class="close" data-dismiss="alert" href="#">&times;</a>
```

Methods

```
$.alert()
```

To enable all alerts to be able to be closed, add the above method. To enable alerts to animate out when closed, make sure they have the `.fade` and `.in` class already applied to them.

Close

Closes an alert.

```
$(".alert").alert('close')
```

Events

There are two event that can be tied to Bootstrap's alert class.

Event	Description
close	This event fires immediately when the close instance method is called.
closed	This event is fired when the alert has been closed (will wait for css transitions to complete).

```
$('#my-alert').bind('closed', function () {  
  // do something..  
})
```

Buttons

Buttons were introduced in chapter 3, and you don't need to do anything to make them work as links, and as buttons in forms. There is some additional interaction that you can add with the plugin, notably loading states, and adding toolbar like functionality to button groups.

Loading State

To add a loading state to a button, simply add `data-loading-text="Loading..."` as an attribute to the button. When the button is clicked, the `.disabled` class is added, giving the appearance that it can no longer be clicked.



Figure 4-6. Loading Button

```
<button type="button" class="btn btn-primary" data-loading-text="Loading...">Submit!</button>
```

Single Toggle

When clicking on a button with the `data-toggle="button"` attribute, a class of `.active` is added.



Figure 4-7. Toggle Button

```
<button type="button" class="btn btn-primary" data-toggle="button">Toggle</button>
```

Checkbox Buttons

Buttons can work like checkboxes, where you can select many of the options in a button group. To add this function, add `data-toggle="buttons-checkbox"` for checkbox style toggling on `btn-group`.



Figure 4-8. Checkbox Buttons

```
<div class="btn-group" data-toggle="buttons-checkbox">  
  <button type="button" class="btn btn-primary">Left</button>  
  <button type="button" class="btn btn-primary">Middle</button>  
  <button type="button" class="btn btn-primary">Right</button>  
</div>
```

Radio Buttons

Radio buttons function similarly to checkboxes, the primary difference is that a radio button doesn't allow for multiple selections, only one in the group. To add this function, add `data-toggle="buttons-radio"` for radio style toggling on `btn-group`.



Figure 4-9. Radio Buttons

```
<div class="btn-group" data-toggle="buttons-radio">
  <button type="button" class="btn btn-primary">Left</button>
  <button type="button" class="btn btn-primary">Middle</button>
  <button type="button" class="btn btn-primary">Right</button>
</div>
```

Usage

The `.button` method can be applied to any class or ID that you want. To enable all buttons in the `.nav-tabs` via Javascript, add this code:

```
$('.nav-tabs').button()
```

Methods

Toggle

Toggles push state. Gives the button the appearance that it has been activated.

```
$('.button('toggle')
```

Loading

Sets button state to loading - disables button and swaps text to loading text. Loading text should be defined on the button element using the data attribute `data-loading-text`.

```
<button type="button" class="btn" data-loading-text="loading stuff..." >...</button>
```

Reset

Resets button state, bringing the original content back to the text. Useful when you need to return the button back to the primary state.

```
$('.button('reset')
```

String

String in this method is referring to any string declared by the user.

```
$(.button\('string'\))
```

To reset the button state, and bring in new content, use the string method.

```
<button type="button" class="btn" data-complete-text="finished!" >...</button>  
  
<script>  
  \$\('.btn'\).button\('complete'\)  
</script>
```

Collapse

The collapse plugin makes it easy to make collapsing divs. Whether you use it to build accordian navigation or content boxes, it allows for a lot of content options.

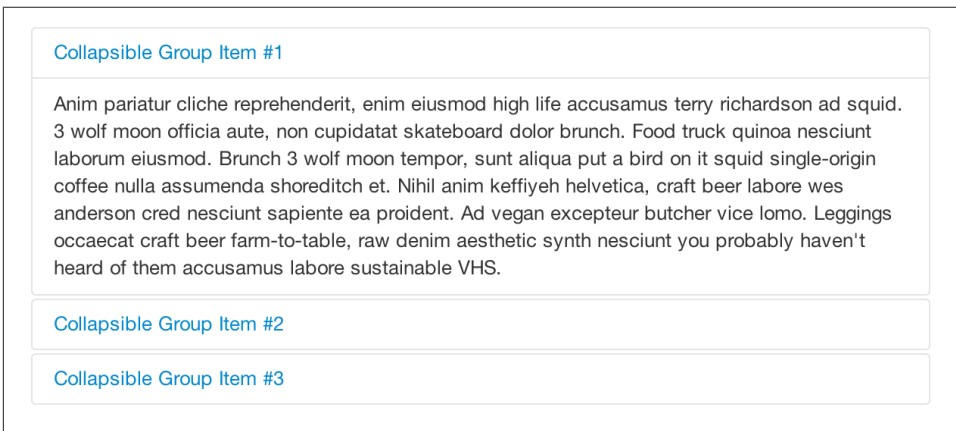


Figure 4-10. Example Accordian

Accordian Code.

```
<div class="accordion" id="accordion2">  
  <div class="accordion-group">  
    <div class="accordion-heading">  
      <a class="accordion-toggle" data-toggle="collapse" data-parent="#accordion2" href="#collapseOne">  
        Collapsible Group Item #1  
      </a>  
    </div>  
    <div id="collapseOne" class="accordion-body collapse in">  
      <div class="accordion-inner">  
        Anim pariatur cliche...  
      </div>  
    </div>  
  </div>  
  <div class="accordion-group">  
    <div class="accordion-heading">
```

```

    <a class="accordion-toggle" data-toggle="collapse" data-parent="#accordion2" href="#collapseTwo">
      Collapsible Group Item #2
    </a>
  </div>
  <div id="collapseTwo" class="accordion-body collapse">
    <div class="accordion-inner">
      Anim pariatur cliche...
    </div>
  </div>
</div>
...

```

You can also use the data attributes to make all content collapsible.

```

<button type="button" class="btn btn-danger" data-toggle="collapse" data-target="#demo">
  simple collapsible
</button>

<div id="demo" class="collapse in"> ... </div>

```

Usage

Via Data Attributes

Like all of the plugins that use the data attributes API, you can add all needed markup without writing any Javascript. 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, add the additional class `.in`.

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

Via Javascript

```
$(".collapse").collapse()
```

Options

Options can be passed via data attributes, or with Javascript.

Name	type	default	description
parent	selector	false	If selector then all collapsible elements under the specified parent will be closed when this collapsible item is shown. (similar to traditional accordion behavior)
toggle	boolean	true	Toggles the collapsible element on invocation

Methods

Options

Activates your content as a collapsible element. Accepts an optional options object.

```
.collapse(options)
```

Toggle

Toggles a collapsible element to shown or hidden.

```
$('#myCollapsible').collapse({  
  toggle: false  
})  
.collapse('toggle')
```

Show

Shows a collapsible element.

```
.collapse('show')
```

Hide

Hides a collapsible element.

```
.collapse('hide')
```

Events

There are four events that can be hooked into with the collapse plugin.

Event	Description
show	This event fires immediately when the <code>show</code> instance method is called.
shown	This event is fired when a collapse element has been made visible to the user (will wait for css transitions to complete).
hide	This event is fired immediately when the <code>hide</code> method has been called.
hidden	This event is fired when a collapse element has been hidden from the user (will wait for css transitions to complete).

```
$('#myCollapsible').on('hidden', function () {  
  // do something...  
})
```

Carousel

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, video, or likely anytime of content that you might want.

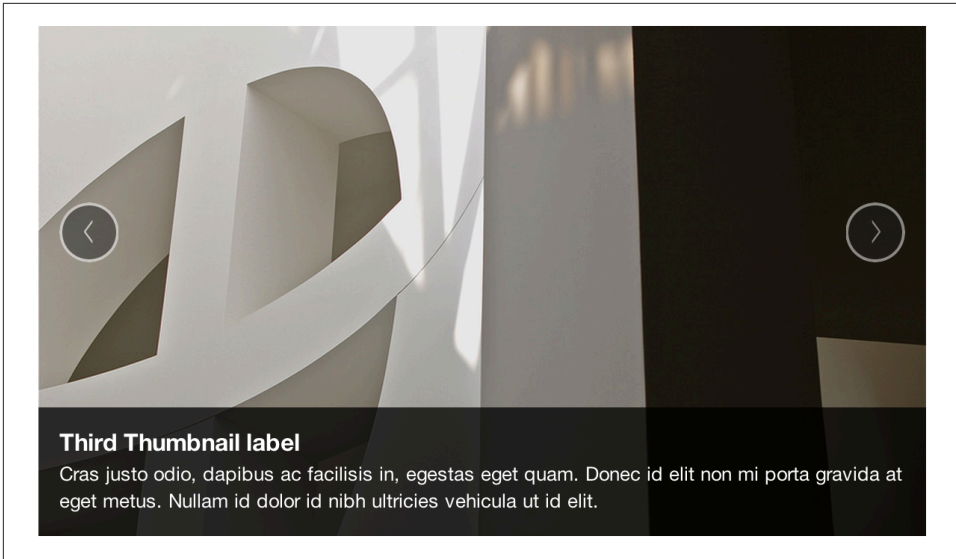


Figure 4-11. Carousel Example

Carousel Code Example.

```
<div id="myCarousel" class="carousel slide">
  <!-- Carousel items -->
  <div class="carousel-inner">
    <div class="active item">...</div>
    <div class="item">...</div>
    <div class="item">...</div>
  </div>
  <!-- Carousel nav -->
  <a class="carousel-control left" href="#myCarousel" data-slide="prev">&lsaquo;</a>
  <a class="carousel-control right" href="#myCarousel" data-slide="next">&rsaquo;</a>
</div>
```

Usage

To implement the carousel, you just need to add the code with the markup above. No need for data attributes, just simple class based development. To call the carousel with Javascript, you can do it manually with the following code:

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

Options

Options can be passed through data attributes, or through Javascript.

Name	type	default	description
interval	number	5000	The amount of time to delay between automatically cycling an item. If false, carousel will not automatically cycle.
pause	string	"hover"	Pauses the cycling of the carousel on mouseenter and resumes the cycling of the carousel on mouseleave.

Methods

Options

Initializes the carousel with an optional options object and starts cycling through items.

```
$('.carousel').carousel({  
  interval: 2000  
})
```

Cycle

Cycles through the carousel items from left to right.

```
.carousel('cycle')
```

Pause

Stops the carousel from cycling through items.

```
.carousel('pause')
```

Number

Cycles the carousel to a particular frame (0 based, similar to an array).

```
.carousel(number)
```

Prev

Cycles to the previous item.

```
.carousel('prev')
```

Next

Cycles to the next item.

```
.carousel('next')
```

Events

The carousel has two events that you can hook into.

Event	Description
slide	This event fires immediately when the slide instance method is invoked.
slid	This event is fired when the carousel has completed its slide transition.

Typeahead

Typeahead allows you to easily create typeahead inputs in forms. Example, you could preload states in a state field, or using some Javascript, get search results using some ajax calls.



Figure 4-12. Typeahead Example

Usage

Using data API, you can add sources via the `data-source` attribute. Items should be listed in either a JSON array, or a function.

Typeahead Code Example.

```

<input
  type="text"
  class="span3"
  data-provide="typeahead"
  data-items="4"
  data-source="[
    'Alabama',
    'Alaska',
    'Arizona',
    'Arkansas',
    'California',
    ...
  ]"
>

```

To call directly with Javascript, use the following method.

Javascript Method.

```

$('.typeahead').typeahead()

```

Options

Name	type	default	description
source	array, function	[]	The data source to query against. May be an array of strings or a function. The function is passed two arguments, the query value in the input field and the process callback. The function may be used synchronously by returning the data source directly or asynchronously via the process callback's single argument.
items	number	8	The max number of items to display in the dropdown.
minLength	number	1	The minimum character length needed before triggering autocomplete suggestions
matcher	function	case insensitive	The method used to determine if a query matches an item. Accepts a single argument, the item against which to test the query. Access the current query with <code>this.query</code> . Return a boolean true if query is a match.
sorter	function	exact match, case sensitive, case insensitive	Method used to sort autocomplete results. Accepts a single argument items and has the scope of the typeahead instance. Reference the current query with <code>this.query</code> .
updater	function	returns selected item	The method used to return selected item. Accepts a single argument, the item and has the scope of the typeahead instance.
highlighter	function	highlights all default matches	Method used to highlight autocomplete results. Accepts a single argument item and has the scope of the typeahead instance. Should return html.

Affix

The affix plugin allows you to allow a `div` to become affixed to a location on the page. A common example of this is social icons on a page. They will start in a location, but as the page hits a certain mark, the `div` will become locked in place and will stop scrolling with the rest of the page.

Usage

To apply the affix plugin to a `div`, you can use either data attributes, or use Javascript directly. Of note, you must position the element so that it can be affixed to the page. Position is controlled by the `data-spy` attribute, using either `affix`, `affix-top`, or `affix-bottom`. You then use the `data-offset` to calculate the position of the scroll.

```
<div data-spy="affix" data-offset-top="200">  
  ...  
</div>
```

Options

Name	type	default	description
offset	number/ function/ object	10	Pixels to offset from screen when calculating position of scroll. If a single number is provided, the offset will be applied in both top and left directions. To listen for a single direction, or multiple unique offsets, just provide an object of <code>offset: { x: 10 }</code> . Use a function when you need to dynamically provide an offset (useful for some responsive designs).

Github Project

Like a lot of great open source projects, the power of Bootstrap comes not just from the developers at the core of the project, but also from the development community that supports it. (GitHub)[<http://github.com>] is a large code repository for projects, and at time of writing, Bootstrap is the most popular project. With over 42,000 stars, and over 10,000 forks, the project is bustling with activity. Like I mentioned in chapter 1, if you want to use Bootstrap, you can simply download the .zip archive from the site, or you can download using `git`.

For the uninitiated, `git` is a free and open source version control system. Bootstrap, and a host of other projects manage everything using Github, which is an online code repository for `git` projects. To download the source for Bootstrap, run the following commands from the command line.

```
$ git clone https://github.com/twitter/bootstrap.git
```

Cloning the Bootstrap will give you a full download of all the files, not just the CSS/Javascript but also all of the documentation pages, and the LESS files for the dynamic Javascript elements.

If desired, using the LESS files, you can compile your own version of Bootstrap with the features, or customizations that you desire.

Customizing Bootstrap

You can download the source, or if you want to easily customize a few of the colors, sizing, or plugins, you can cater the Bootstrap to your needs via the [Bootstrap website] (<http://twitter.github.com/bootstrap/customize.html>).

Scaffolding	Typography	Form states & alerts
@bodyBackground <input type="text" value="@white"/>	@sansFontFamily <input type="text" value="'Helvetica Neue', Helvetica, Aria"/>	@warningText <input type="text" value="#c09853"/>
@textColor <input type="text" value="@grayDark"/>	@serifFontFamily <input type="text" value="Georgia, 'Times New Roman', Times"/>	@warningBackground <input type="text" value="#fcf8e3"/>
Links	@monoFontFamily <input type="text" value="Menlo, Monaco, 'Courier New', mon"/>	@errorText <input type="text" value="#b94a48"/>
@linkColor <input type="text" value="#08c"/>	@baseFontSize <input type="text" value="14px"/>	@errorBackground <input type="text" value="#f2dede"/>
@linkColorHover <input type="text" value="darken(@linkColor, 15%)"/>	@baseFontFamily <input type="text" value="@sansFontFamily"/>	@successText <input type="text" value="#468847"/>
Colors	@baseLineHeight <input type="text" value="20px"/>	@successBackground <input type="text" value="#dff0d8"/>
@blue <input type="text" value="#049cdb"/>	@altFontFamily <input type="text" value="@serifFontFamily"/>	@infoText <input type="text" value="#3a87ad"/>

Figure 5-1. Customize Page

When you use the customize page, you decide what components you need, you might choose to leave off all of the responsive features, or maybe leave off the button classes if they conflict with styles that you already have. Then you can opt out of any of the jQuery plugins. If you know that you aren't going to be using the modals, or the carousel, you could leave it out of the build so that you a smaller file to request.

Lastly, you have the LESS variables that you can configure. Everything from column count, to typography colors can be modified here. When you have the options included that you can download a customized build of Bootstrap.

Using LESS

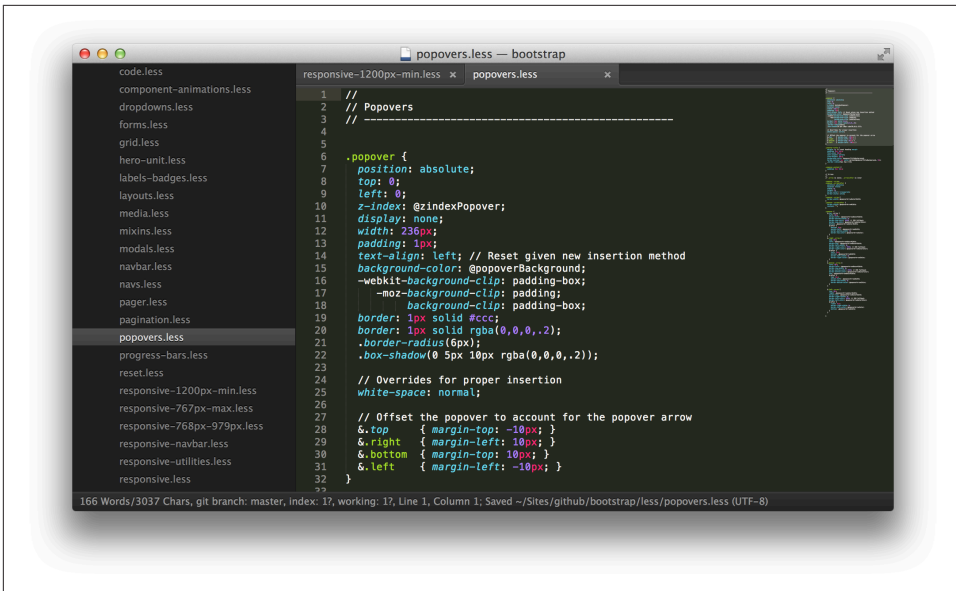


Figure 5-2. Using Bootstrap with LESS

There are a few different ways to use LESS with Bootstrap. The first, and perhaps the easiest is to use a preprocessor like Codekit, or SimpLess. Using these tools, you can set them up to watch certain files or folders. Then when you save any of those files, they build the master CSS files. In addition to using traditional CSS techniques, you can use advanced features like mixins and functions to dynamically change the look of your site just by changing some variables.

LESS is a dynamic stylesheet language for writing CSS. It allows you to write variables, functions, and mixins for your CSS. The Bootstrap /less/ folder has a few files, but for setting globals, check the `variables.less` file and `mixins.less`.

In the `variables.less` file, you will find all of the global variables for Bootstrap. Let's say you wanted to change the color of all of the links. You would simply update the following, update LESS to compile the CSS, and all of the links would change color.

```

/* Old Code */
@linkColor:                #08c;
@linkColorHover:           darken(@linkColor, 15%);

/* New Code */
@linkColor:                #7d00cc;
@linkColorHover:           darken(@linkColor, 15%);

```

Now, all links are changed to a purple color, and buttons, and buttons and other interface elements that call for `@linkColor` will be updated all over your site.

In the beginning, I was really hesitant to use LESS, after all, I have been writing CSS for a long time, and didn't feel the need to change. The nesting alone is such a huge timesaver, I'm really glad to have added it to my workflow.

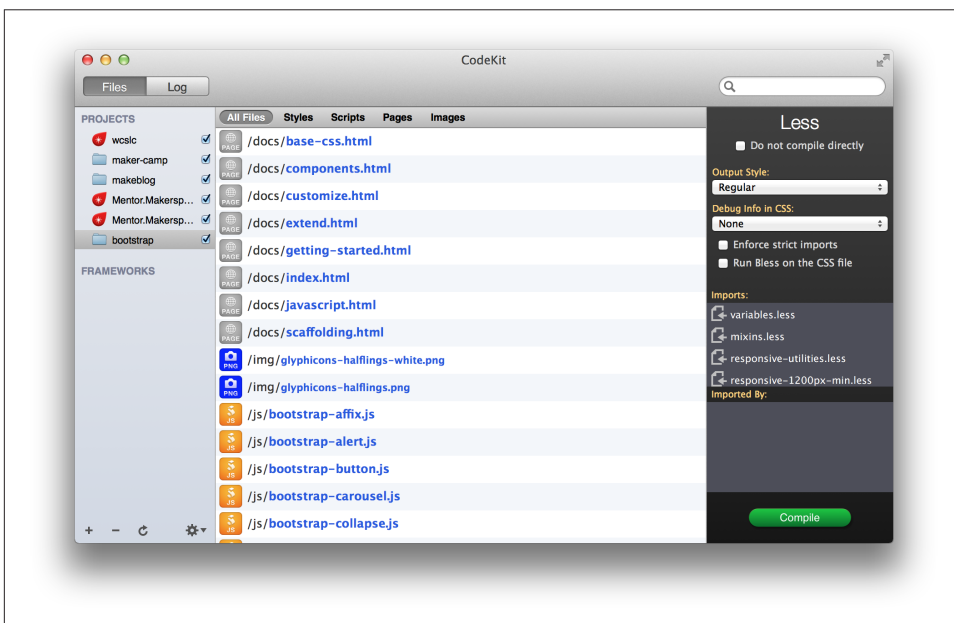


Figure 5-3. Codekit and the Bootstrap code base

Text Snippets

To rapidly develop Bootstrap sites, I like to use [Sublime Text 2](#), and the Bootstrap snippets from [DEVtellect](#) on [Github](#). This makes any component easy as a keyboard shortcut to add to the page.

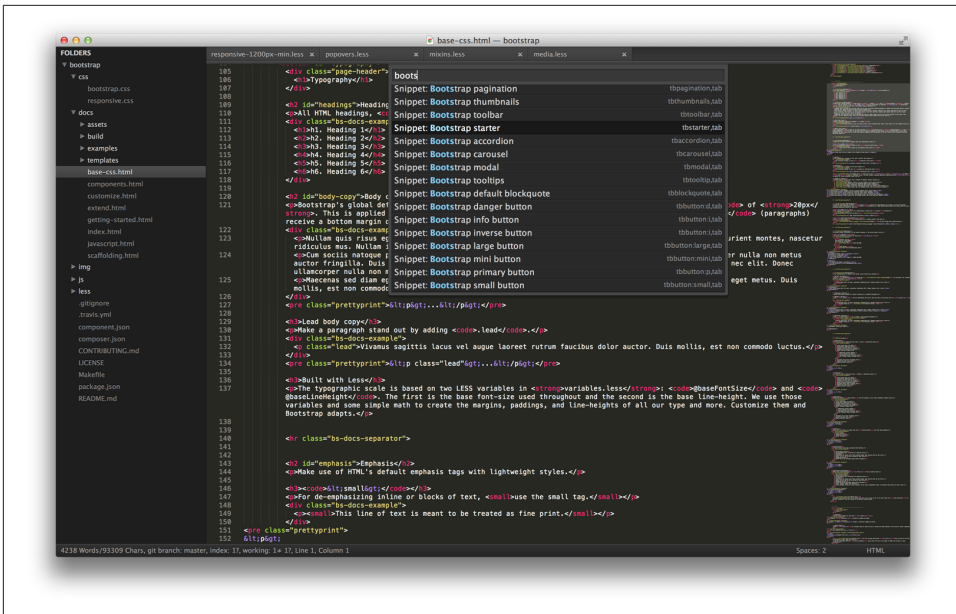


Figure 5-4. Bootstrap Snippets

To install the snippets, clone the git repository into your packages folder.

```
git clone git@github.com:devtlect/sublime-twitter-bootstrap-snippets.git
```

There are clippings for lots of popular IDEs/Text Editors.

Photoshop Templates

In addition to snippets, there have been a few Photoshop PSDs of all of the Bootstrap markup elements. My favorite comes from [Repix design](#).



Features:

- Adjustable colors
- Separate layers
- Vector based

The PSD, is free, but the author requests that you pay with a tweet. Kind a new spin on the free as in free speech mantra of Open Source programming.

Themes

Built with Bootstrap

If you are looking for even more inspiration, check out [Built with Bootstrap](#), a Tumblr that features screen grabs of user-submitted Bootstrap sites. Fun way to see the varying ways that people are using Bootstrap.

Conclusion

So, to conclude, whatever the project, Bootstrap can fill the needs of just about any web project. It's blend of responsive framework, extensive Javascript plugins and robust in-

terface components make developing easy, fast, and feature rich. It has been great working with the project over the last year, and I look forward to the future development of the project. Cheers, kudos, and all of the accolades to Jacob Thornton and Mark Otto for creating a project that is so versatile, and fun to develop around.

Thanks from me,

Jake Spurlock

About the Authors

Jake Spurlock is a developer for O'Reilly Media where he works for MAKE magazine. MAKE publishes a DIY magazine, produces Maker Faire, and is trying to make the world a better place by teaching people that they can make things. Jake is a Utah native, but a year and a half ago was grafted into the California wine country.

Colophon

The animal on the cover of *FILL IN TITLE* is FILL IN DESCRIPTION.

The cover image is from FILL IN CREDITS. The cover font is Adobe ITC Garamond. The text font is Adobe Minion Pro; the heading font is Adobe Myriad Condensed; and the code font is Dalton Maag's Ubuntu Mono.

We'd like to hear your suggestions for improving our indexes. Send email to index@oreilly.com.