



Community Experience Distilled

# Responsive Web Design with HTML5 and CSS3

*Second Edition*

Build responsive and future-proof websites to meet the demands of modern web users

Ben Frain

**[PACKT]**  
PUBLISHING

# Responsive Web Design with HTML5 and CSS3

*Second Edition*

Build responsive and future-proof websites to meet  
the demands of modern web users

**Ben Frain**



BIRMINGHAM - MUMBAI

# Responsive Web Design with HTML5 and CSS3

## *Second Edition*

Copyright © 2015 Packt Publishing

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author, nor Packt Publishing, and its dealers and distributors will be held liable for any damages caused or alleged to be caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

First published: April 2012

Second edition: August 2015

Production reference: 2200815

Published by Packt Publishing Ltd.

Livery Place

35 Livery Street

Birmingham B3 2PB, UK.

ISBN 978-1-78439-893-4

[www.packtpub.com](http://www.packtpub.com)

# Credits

**Author**

Ben Frain

**Reviewers**

Esteban S. Abait

Christopher Scott Hernandez

Mauvis Ledford

Sophie Williams

**Commissioning Editor**

Edward Gordon

**Acquisition Editors**

Edward Gordon

Subho Gupta

**Content Development Editor**

Pooja Nair

**Technical Editor**

Ankita Thakur

**Copy Editors**

Rebecca Youé

Sonia Cheema

**Project Coordinator**

Bijal Patel

**Proofreader**

Safis Editing

**Indexer**

Mariammal Chettiyar

**Production Coordinator**

Nilesh R. Mohite

**Cover Work**

Nilesh R. Mohite

# About the Author

**Ben Frain** has been a web designer/developer since 1996. He is currently employed as a Senior Front-end Developer at Bet365.

Before the web, he worked as an underrated (and modest) TV actor and technology journalist, having graduated from Salford University with a degree in Media and Performance.

He has written four equally underrated (his opinion) screenplays and still harbors the (fading) belief he might sell one. Outside of work, he enjoys simple pleasures. Playing indoor football while his body and wife still allow it, and wrestling with his two sons.

His other book, *Sass and Compass for Designers* is available now. Visit Ben online at [www.benfrain.com](http://www.benfrain.com) and follow him on Twitter at [twitter.com/benfrain](https://twitter.com/benfrain).

---

I'd like to thank the technical reviewers of this book for giving up their free time to provide valuable input. Thanks to them, this is a better product.

I'd also like to thank the web community at large for their continued sharing of information. Without them, I wouldn't be able to enjoy my working days as a web developer.

Most importantly, a note of appreciation for my family. Many episodes of sub-standard TV (wife), cups of tea (parents), and piratical sword-fights (sons) were sacrificed for the writing of this book.

---

# About the Reviewers

**Esteban S. Abait** is a senior software architect and former PhD student. He has experience devising the architecture of complex software products, and planning their development. He has worked both onsite and offshore for clients such as Cisco, Intuit, and Southwest. Throughout his career, he has worked with different technologies such as Java, PHP, Ruby, and Node.js among others. In recent years, his main interests have revolved around web, mobile and REST APIs. He has developed large, maintainable web applications using JavaScript. In addition, he has worked to assess clients on REST best practices. On the other hand, he has worked on high traffic websites, where topics such as replication, sharding, or distributed caches are key to scalability.

Esteban is currently working at Globant as a technical director. In this role, he works to ensure projects' delivery meet their deadlines with the best quality. He also designs software program training, and interviews software developers. In addition, he usually travels to clients to provide consultancy on web technologies.

Globant (<http://www.globant.com/>) is a new breed of technology service provider, focused on delivering innovative software solutions by leveraging emerging technologies and trends. Globant combines the engineering and technical rigor of IT service providers with the creative and cultural approach of digital agencies. Globant is the place where engineering, design, and innovation meet scale.

**Christopher Scott Hernandez** is a designer turned developer who has been working on the Web since 1996, when he built the Web's first boat upholstery site for his dad. He's since moved on to bring his expertise to companies small and large, having worked on some of the most visited sites in the world including eBay, LinkedIn, and Apple.

He was also a technical reviewer for *HTML5 Multimedia Development Cookbook*, Packt Publishing. Chris is an avid reader and lover of books. When he's not pushing pixels and writing code, he enjoys spending time with his wife and daughter exploring the parks and trails of the beautiful Austin, Texas.

**Mauvis Ledford** is a full-stack founder and CTO specializing in the realm of the web, mobile web, and scaling applications on the cloud.

Mauvis has contributed to products at Disney Mobile, Skype, Netflix, and many start-ups in the San Francisco and New York City areas. He is currently CTO at Pathbrite, an EdTech start-up specializing in free, responsive, multimedia e-portfolios and digital resumes for everyone. Create your own at <http://www.pathbrite.com>.

Mauvis was also a technical reviewer for the first edition of *Responsive Web Design with HTML5 and CSS3*, Packt Publishing and *Building Hybrid Android Apps with Java and JavaScript*, O'Reilly Media.

**Sophie Williams** is a bit of a perfectionist and has a thing for typography. She has a degree in graphic design and is currently a web/UI designer at [www.bet365.com](http://www.bet365.com). While she loves designing for the Web, she will always have a special place in her heart for letterpress and print. Outside of work, she makes mean cupcakes, experiments with arts and crafts, and loves to point out (to anyone who will listen) when anything in the real world is misaligned.

You can find Sophie at [www.sophiewill.com](http://www.sophiewill.com) or follow her on Twitter @sophiewill113.

# www.PacktPub.com

## Support files, eBooks, discount offers, and more

For support files and downloads related to your book, please visit [www.PacktPub.com](http://www.PacktPub.com).

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at [www.PacktPub.com](http://www.PacktPub.com) and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at [service@packtpub.com](mailto:service@packtpub.com) for more details.

At [www.PacktPub.com](http://www.PacktPub.com), you can also read a collection of free technical articles, sign up for a range of free newsletters, and receive exclusive discounts and offers on Packt books and eBooks.



<https://www2.packtpub.com/books/subscription/packtlib>

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can search, access, and read Packt's entire library of books.

## Why subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print, and bookmark content
- On demand and accessible via a web browser

## Free access for Packt account holders

If you have an account with Packt at [www.PacktPub.com](http://www.PacktPub.com), you can use this to access PacktLib today and view 9 entirely free books. Simply use your login credentials for immediate access.



# Table of Contents

<b>Preface</b>	<b>ix</b>
<b>Chapter 1: The Essentials of Responsive Web Design</b>	<b>1</b>
<b>Beginning our quest</b>	<b>2</b>
<b>Defining responsive web design</b>	<b>2</b>
Responsive web design in a nutshell	2
<b>Setting browser support levels</b>	<b>3</b>
A brief note on tooling and text editors	5
<b>Our first responsive example</b>	<b>5</b>
Our basic HTML file	6
Taming images	9
Enter media queries	12
Amending the example for a larger screen	13
<b>The shortcomings of our example</b>	<b>17</b>
<b>Summary</b>	<b>17</b>
<b>Chapter 2: Media Queries – Supporting Differing Viewports</b>	<b>19</b>
<b>Why media queries are needed for a responsive web design</b>	<b>20</b>
Basic conditional logic in CSS	21
<b>Media query syntax</b>	<b>21</b>
Media queries in link tags	22
<b>Combining media queries</b>	<b>23</b>
Media queries with @import	24
Media queries in CSS	24
What can media queries test for?	25
<b>Using media queries to alter a design</b>	<b>26</b>
Any CSS can be wrapped in a media query	28
Media queries for HiDPI devices	28
<b>Considerations for organizing and authoring media queries</b>	<b>29</b>
Linking to different CSS files with media queries	29

The practicalities of separating media queries	30
Nesting media queries 'inline'	30
<b>Combine media queries or write them where it suits?</b>	<b>31</b>
<b>The viewport meta tag</b>	<b>33</b>
<b>Media Queries Level 4</b>	<b>35</b>
Scripting media feature	35
Interaction media features	36
The hover media feature	37
Environment media features	38
<b>Summary</b>	<b>38</b>
<b>Chapter 3: Fluid Layouts and Responsive Images</b>	<b>39</b>
<b>Converting a fixed pixel design to a fluid proportional layout</b>	<b>40</b>
Why do we need Flexbox?	45
Inline block and whitespace	45
Floats	45
Table and table-cell	46
<b>Introducing Flexbox</b>	<b>46</b>
The bumpy path to Flexbox	47
Browser support for Flexbox	47
Leave prefixing to someone else	47
<b>Getting Flexy</b>	<b>49</b>
Perfect vertically centered text	49
Offset items	50
Reverse the order of items	52
How about if we want them laid out vertically instead?	53
Column reverse	53
Different Flexbox layouts inside different media queries	53
Inline-flex	54
Flexbox alignment properties	55
The align-items property	57
The align-self property	58
Possible alignment values	59
The justify-content property	59
The flex property	61
Simple sticky footer	64
Changing source order	65
Wrapping up Flexbox	71
<b>Responsive images</b>	<b>71</b>
The intrinsic problem of responsive images	71
Simple resolution switching with srcset	72
Advanced switching with srcset and sizes	73
Did you say the browser 'might' pick one image over another?	74

---

Art direction with the picture element	74
Facilitate new-fangled image formats	75
<b>Summary</b>	<b>76</b>
<b>Chapter 4: HTML5 for Responsive Web Designs</b>	<b>77</b>
<b>HTML5 markup – understood by all modern browsers</b>	<b>78</b>
<b>Starting an HTML5 page the right way</b>	<b>79</b>
The doctype	79
The HTML tag and lang attribute	80
Specifying alternate languages	80
Character encoding	81
<b>Easy-going HTML5</b>	<b>81</b>
A sensible approach to HTML5 markup	82
All hail the mighty <a> tag	83
<b>New semantic elements in HTML5</b>	<b>83</b>
The <main> element	84
The <section> element	85
The <nav> element	85
The <article> element	86
The <aside> element	86
The <figure> and <figcaption> elements	86
The <details> and <summary> elements	87
The <header> element	89
The <footer> element	89
The <address> element	90
A note on h1-h6 elements	90
<b>HTML5 text-level semantics</b>	<b>91</b>
The <b> element	91
The <em> element	91
The <i> element	91
<b>Obsolete HTML features</b>	<b>92</b>
<b>Putting HTML5 elements to use</b>	<b>93</b>
Applying common sense to your element selection	94
<b>WCAG and WAI-ARIA for more accessible web applications</b>	<b>94</b>
WCAG	94
WAI-ARIA	95
Don't use roles for semantic elements	95
If you only remember one thing	96
Taking ARIA further	96
<b>Embedding media in HTML5</b>	<b>97</b>
Adding video and audio the HTML5 way	97
Fallback capability for older browsers	99

Audio and video tags work almost identically	99
<b>Responsive HTML5 video and iFrames</b>	<b>99</b>
<b>A note about 'offline first'</b>	<b>101</b>
<b>Summary</b>	<b>102</b>
<b>Chapter 5: CSS3 – Selectors, Typography, Color Modes, and New Features</b>	<b>103</b>
<b>No one knows it all</b>	<b>104</b>
<b>Anatomy of a CSS rule</b>	<b>104</b>
<b>Quick and useful CSS tricks</b>	<b>105</b>
CSS multi-column layouts for responsive designs	105
Fixed columns, variable width	107
Adding a gap and column divider	107
<b>Word wrapping</b>	<b>108</b>
Text ellipsis	109
Creating horizontal scrolling panels	110
<b>Facilitating feature forks in CSS</b>	<b>112</b>
Feature queries	112
Combining conditionals	114
Modernizr	114
Feature detection with Modernizr	115
<b>New CSS3 selectors and how to use them</b>	<b>116</b>
CSS3 attribute selectors	117
CSS3 substring matching attribute selectors	117
The 'beginning with' substring matching attribute selector	118
The 'contains an instance of' substring matching attribute selector	118
The 'ends with' substring matching attribute selector	119
Gotchas with attribute selection	119
Attribute selectors allow you to select IDs and classes that start with numbers	120
<b>CSS3 structural pseudo-classes</b>	<b>121</b>
The :last-child selector	121
The nth-child selectors	122
Understanding what nth rules do	122
Breaking down the math	123
nth-based selection in responsive web designs	125
The negation (:not) selector	128
The empty (:empty) selector	129
Do something with the :first-line regardless of viewport	130
<b>CSS custom properties and variables</b>	<b>130</b>
<b>CSS calc</b>	<b>131</b>
<b>CSS Level 4 selectors</b>	<b>131</b>
The :has pseudo class	132

---

Responsive viewport-percentage lengths (vmax, vmin, vh, vw)	132
<b>Web typography</b>	<b>133</b>
The @font-face CSS rule	133
Implementing web fonts with @font-face	134
A note about custom @font-face typography and responsive designs	136
<b>New CSS3 color formats and alpha transparency</b>	<b>137</b>
RGB color	137
HSL color	138
Alpha channels	139
Color manipulation with CSS Color Module Level 4	140
<b>Summary</b>	<b>140</b>
<b>Chapter 6: Stunning Aesthetics with CSS3</b>	<b>141</b>
<b>Text shadows with CSS3</b>	<b>142</b>
Omitting the blur value when not needed	143
Multiple text shadows	143
<b>Box shadows</b>	<b>144</b>
An inset shadow	144
Multiple shadows	145
Understanding spread	145
<b>Background gradients</b>	<b>146</b>
The linear-gradient notation	147
Specifying gradient direction	147
Color stops	148
Adding fallback for older browsers	149
Radial background gradients	149
Breakdown of the radial-gradient syntax	150
Handy 'extent' keywords for responsive sizing	150
<b>Repeating gradients</b>	<b>151</b>
<b>Background gradient patterns</b>	<b>152</b>
<b>Multiple background images</b>	<b>154</b>
Background size	155
Background position	155
Background shorthand	156
<b>High-resolution background images</b>	<b>157</b>
<b>CSS filters</b>	<b>158</b>
Available CSS filters	159
Combining CSS filters	164
<b>A warning on CSS performance</b>	<b>165</b>
A note on CSS masks and clipping	166
<b>Summary</b>	<b>167</b>

<b>Chapter 7: Using SVGs for Resolution Independence</b>	<b>169</b>
<b>A brief history of SVG</b>	<b>171</b>
<b>The graphic that is a document</b>	<b>172</b>
The root SVG element	173
Namespace	174
The title and desc tags	174
The defs tag	174
The g element	175
SVG shapes	175
SVG paths	175
<b>Creating SVGs with popular image editing packages and services</b>	<b>176</b>
Save time with SVG icon services	176
<b>Inserting SVGs into your web pages</b>	<b>177</b>
Using an img tag	178
Using an object tag	178
Insert an SVG as a background image	179
A brief aside on data URIs	180
Generating image sprites	181
<b>Inserting an SVG inline</b>	<b>182</b>
Re-using graphical objects from symbols	183
Inline SVGs allow different colors in different contexts	185
Make dual-tone icons that inherit the color of their parent	185
Re-using graphical objects from external sources	186
<b>What you can do with each SVG insertion method (inline, object, background-image, and img)</b>	<b>187</b>
Browser schisms	188
<b>Extra SVG capabilities and oddities</b>	<b>188</b>
SMIL animation	188
The end of SMIL	189
Styling an SVG with an external style sheet	190
Styling an SVG with internal styles	191
SVG properties and values within CSS	191
Animate an SVG with CSS	191
<b>Animating SVG with JavaScript</b>	<b>193</b>
A simple example of animating an SVG with GreenSock	194
<b>Optimising SVGs</b>	<b>196</b>
<b>Using SVGs as filters</b>	<b>196</b>
<b>A note on media queries inside SVGs</b>	<b>199</b>
Implementation tips	200
Further resources	201
<b>Summary</b>	<b>201</b>

---

<b>Chapter 8: Transitions, Transformations, and Animations</b>	<b>203</b>
<b>What CSS3 transitions are and how we can use them</b>	<b>204</b>
The properties of a transition	206
The transition shorthand property	207
Transition different properties over different periods of time	208
Understanding timing functions	208
Fun transitions for responsive websites	210
<b>CSS3 2D transforms</b>	<b>210</b>
Scale	211
Translate	212
Using translate to center absolutely positioned elements	212
Rotate	214
Skew	215
Matrix	215
Matrix transformations for cheats and dunces	216
The transform-origin property	217
<b>CSS3 3D transformations</b>	<b>218</b>
The transform3d property	221
Use transforms with progressive enhancement	222
<b>Animating with CSS3</b>	<b>225</b>
The animation-fill-mode property	228
<b>Summary</b>	<b>230</b>
<b>Chapter 9: Conquer Forms with HTML5 and CSS3</b>	<b>231</b>
<b>HTML5 forms</b>	<b>232</b>
<b>Understanding the component parts of HTML5 forms</b>	<b>233</b>
placeholder	234
Styling the placeholder text	234
required	234
autofocus	236
autocomplete	236
List and the associated datalist element	237
<b>HTML5 input types</b>	<b>239</b>
email	239
number	240
min and max ranges	241
Changing the step increments	241
url	242
tel	243
search	244
pattern	245
color	246

Date and time inputs	246
date	247
month	247
week	248
time	249
range	249
<b>How to polyfill non-supporting browsers</b>	<b>250</b>
<b>Styling HTML5 forms with CSS3</b>	<b>251</b>
Indicating required fields	254
Creating a background fill effect	256
<b>Summary</b>	<b>257</b>
<b>Chapter 10: Approaching a Responsive Web Design</b>	<b>259</b>
<b>Get designs in the browser as soon as possible</b>	<b>260</b>
Let the design dictate the breakpoints	260
<b>View and use the design on real devices</b>	<b>261</b>
<b>Embracing progressive enhancement</b>	<b>262</b>
<b>Defining a browser support matrix</b>	<b>263</b>
Functional parity, not aesthetic parity	263
Choosing the browsers to support	263
<b>Tiering the user experience</b>	<b>264</b>
Practically delivering experience tiers	264
<b>Linking CSS breakpoints to JavaScript</b>	<b>265</b>
<b>Avoid CSS frameworks in production</b>	<b>267</b>
<b>Coding pragmatic solutions</b>	<b>268</b>
When a link becomes a button	269
<b>Use the simplest code possible</b>	<b>270</b>
<b>Hiding, showing, and loading content across viewports</b>	<b>271</b>
Let CSS do the (visual) heavy lifting	272
<b>Validators and linting tools</b>	<b>272</b>
<b>Performance</b>	<b>274</b>
<b>The next big things</b>	<b>275</b>
<b>Summary</b>	<b>276</b>
<b>Index</b>	<b>277</b>

---

# Preface

A responsive web design provides a single solution that looks great on a phone, desktop, and everything in-between. It will effortlessly respond to the size of the user's screen, providing the best experience possible for both today's and tomorrow's devices.

This book covers every essential aspect of responsive web design. In addition, it extends the responsive design methodology by applying the latest and most useful techniques provided by HTML5 and CSS3, making designs leaner and more maintainable than ever before. It also explains common best practice methods of writing and delivering code, images, and files.

If you can understand HTML and CSS, you can build a responsive web design.

## What this book covers

*Chapter 1, The Essentials of Responsive Web Design*, is a whistle-stop tour of the key ingredients in coding a responsive web design.

*Chapter 2, Media Queries – Supporting Differing Viewports*, covers everything you need to know about CSS media queries: their capabilities, their syntaxes, and the various ways you can wield them.

*Chapter 3, Fluid Layouts and Responsive Images*, shows you how to code proportional layouts and responsive images, and provides a thorough exploration of Flexbox layouts.

*Chapter 4, HTML5 for Responsive Web Designs*, covers all the semantic elements of HTML5, text-level semantics, and considerations of accessibility. We also cover how to insert video and audio into our pages with HTML5.

*Chapter 5, CSS3 – Selectors, Typography, Color Modes, and New Features*, gets to grips with the endless possibilities of CSS: selectors, HSLA and RGBA colors, web typography, viewport relative units, and a whole lot more.

*Chapter 6, Stunning Aesthetics with CSS3*, covers CSS filters, box shadows, linear and radial gradients, multiple backgrounds, and how to target background images to high-resolution devices.

*Chapter 7, Using SVGs for Resolution Independence*, explains everything we need to use SVGs inside documents and as background images, as well as how to interact with them using JavaScript.

*Chapter 8, Transitions, Transformations, and Animations*, our CSS gets moving as we explore how to make interactions and animations using CSS.

*Chapter 9, Conquer Forms with HTML5 and CSS3*, web forms have always been tough but the latest HTML5 and CSS3 features make them easier to deal with than ever before.

*Chapter 10, Approaching a Responsive Web Design*, explores the essential considerations before embarking on a responsive web design and also provides a few last minute nuggets of wisdom to aid you in your responsive quest.

## What you need for this book

- A text editor
- An evergreen browser
- A penchant for mediocre jokes

## Who this book is for

Are you writing two websites: one for mobile and one for larger displays? Or perhaps you've already implemented your first 'RWD' but are struggling to bring it all together? If so, *Responsive Web Design with HTML5 and CSS3 Second Edition* gives you everything you need to take your websites to the next level.

You'll need some HTML and CSS knowledge to follow along, but everything you need to know about responsive design and making great websites is included in the book!

---

## Conventions


In this book, you will find a number of text styles that distinguish between different kinds of information. Here are some examples of these styles and an explanation of their meaning.


Code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows: "We can fix that prior problem easily by adding this snippet in the `<head>`."

A block of code is set as follows:

```
img {  
    max-width: 100%;  
}
```

**New terms** and **important words** are shown in bold. Words that you see on the screen, for example, in menus or dialog boxes, appear in the text like this: "At its simplest, you pick a URL and click on **START TEST**."

 Warnings or important notes appear in a box like this.

 Tips and tricks appear like this.

## Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book – what you liked or disliked. Reader feedback is important for us as it helps us develop titles that you will really get the most out of.

To send us general feedback, simply e-mail [feedback@packtpub.com](mailto:feedback@packtpub.com), and mention the book's title in the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide at [www.packtpub.com/authors](http://www.packtpub.com/authors).

## Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

## Downloading the example code

You can download the example code files from your account at <http://www.packtpub.com> for all the Packt Publishing books you have purchased. If you purchased this book elsewhere, you can visit <http://www.packtpub.com/support> and register to have the files e-mailed directly to you.

## Downloading the color images of this book

We also provide you with a PDF file that has color images of the screenshots/diagrams used in this book. The color images will help you better understand the changes in the output. You can download this file from: [https://www.packtpub.com/sites/default/files/downloads/89340T\\_ColorImages.pdf](https://www.packtpub.com/sites/default/files/downloads/89340T_ColorImages.pdf).

## Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books – maybe a mistake in the text or the code – we would be grateful if you could report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting <http://www.packtpub.com/submit-errata>, selecting your book, clicking on the **Errata Submission Form** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded to our website or added to any list of existing errata under the Errata section of that title.

To view the previously submitted errata, go to <https://www.packtpub.com/books/content/support> and enter the name of the book in the search field. The required information will appear under the **Errata** section.

## **Piracy**

Piracy of copyrighted material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works in any form on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at [copyright@packtpub.com](mailto:copyright@packtpub.com) with a link to the suspected pirated material.

We appreciate your help in protecting our authors and our ability to bring you valuable content.

## **Questions**

If you have a problem with any aspect of this book, you can contact us at [questions@packtpub.com](mailto:questions@packtpub.com), and we will do our best to address the problem.



# 1

## The Essentials of Responsive Web Design

Only a few years ago, websites could be built at a fixed width, with the expectation that all end users would get a fairly consistent experience. This fixed width (typically 960px wide or thereabouts) wasn't too wide for laptop screens, and users with large resolution monitors merely had an abundance of margin either side.

But in 2007, Apple's iPhone ushered in the first truly usable phone browsing experience, and the way people access and interact with the Web changed forever.

In the first edition of this book, it was noted that:

*"in the 12 months from July 2010 to July 2011, global mobile browser use had risen from 2.86 to 7.02 percent."*

In mid-2015, the same statistics system ([gs.statcounter.com](http://gs.statcounter.com)) reported that this figure had risen to 33.47%. By way of comparison, North America's mobile figure is at 25.86%.

By any metric, mobile device usage is rising ever upwards, while at the other end of the scale, 27 and 30 inch displays are now also commonplace. There is now a greater difference between the smallest and the largest screens browsing the Web than ever before.

Thankfully, there is a solution to this ever-expanding browser and device landscape. A responsive web design, built with HTML5 and CSS3, allows a website to 'just work' across multiple devices and screens. It enables the layout and capabilities of a website to respond to their environment (screen size, input type, device/browser capabilities).

Furthermore, a responsive web design, built with HTML5 and CSS3, can be implemented without the need for server based/back-end solutions.

## Beginning our quest

Whether you're new to responsive web design, HTML5, or CSS3, or already well versed, I'm hoping this first chapter will serve one of two purposes.

If you're already using HTML5 and CSS3 in your responsive web designs, this first chapter should serve as a quick and basic refresher. Alternatively, if you're a newcomer, think of it as a 'boot camp' of sorts, covering the essentials so we're all on the same page.

By the end of this first chapter, we will have covered everything you need to author a fully responsive web page.

You might be wondering why the other nine chapters are here. By the end of this chapter, that should be apparent too.

Here's what we will cover in this first chapter:

- Defining responsive web design
- How to set browser support levels
- A brief discussion on tooling and text editors
- Our first responsive example: a simple HTML5 page
- The importance of the viewport meta tag
- How to make images scale to their container
- Writing CSS3 media queries to create design breakpoints
- The shortfalls in our basic example
- Why our journey has only just begun

## Defining responsive web design

The term, "responsive web design" was coined by Ethan Marcotte in 2010. In his seminal *A List Apart* article (<http://www.alistapart.com/articles/responsive-web-design/>), he consolidated three existing techniques (flexible grid layout, flexible images/media, and media queries) into one unified approach and named it responsive web design.

## Responsive web design in a nutshell

Responsive web design is the presentation of web content in the most relevant format for the viewport and device accessing it.

In its infancy, it was typical for a responsive design to be built starting with the 'desktop', fixed-width design. Content was then reflowed, or removed so that the design worked on smaller screens. However, processes evolved and it became apparent that everything from design, to content and development, worked much better when working in the opposite direction; starting with smaller screens and working up.

Before we get into this, there are a couple of subjects I'd like to address before we continue; browser support and text editors/tooling.

## Setting browser support levels

The popularity and ubiquity of responsive web design makes it an easier sell to clients and stakeholders than ever before. Most people have some idea what responsive web design is about. The notion of a single codebase that will just work across all devices is a compelling offering.

One question that almost always comes up when starting a responsive design project is that of browser support. With so many browser and device variants, it's not always pragmatic to support every single browser permutation fully. Perhaps time is a limiting factor, perhaps money. Perhaps both.

Typically, the older the browser, the greater the work and code required to gain feature or aesthetic parity with modern browsers. Therefore, it may make more sense to have a leaner, and therefore faster, codebase by tiering the experience and only providing enhanced visuals and capabilities for more capable browsers.

In the previous edition of this book, some time was spent covering how to cater for very old desktop-only browsers. In this edition, we will not.

As I write this in mid-2015, Internet Explorer 6, 7, and 8 are all but gone. Even IE 9 only has a 2.45% worldwide share of the browser market (IE 10 is only 1.94% while IE 11 is rising nicely at 11.68%). If you have no alternative but to develop for Internet Explorer 8 and below, you have my sympathies and I'm afraid I must be upfront and advise you that there won't be a terrific amount you can use in this book.

For everyone else, you owe it to your client/paymaster to explain why developing for ailing browsers might be a mistake and investing development time and resource primarily for modern browsers and platforms makes good fiscal sense in every respect.

Ultimately however, the only statistics that really matter are yours. In all but extreme cases, the sites we build should at least be functional in every common browser. Beyond basic functionality, for any web project it makes sense to decide, in advance, what platforms you want to fully enhance the experience for, and which you are happy to concede visual/functional anomalies to.

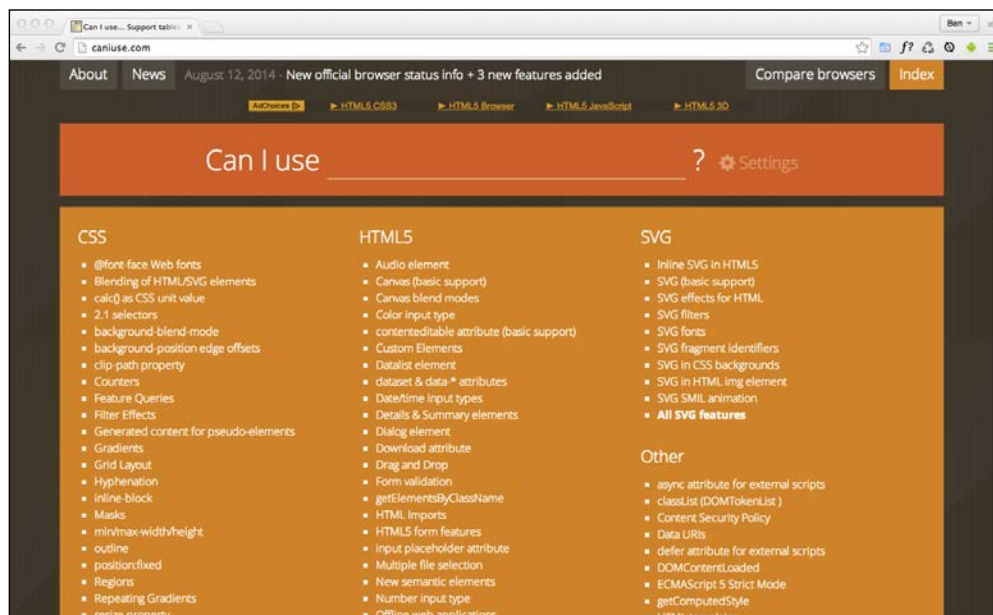
You'll also find that practically, starting with the simplest 'base level' experience and enhancing (an approach known as **progressive enhancement**) is easier than coming at the problem from the opposite direction—building the ultimate experience first then attempting to provide fall backs for less capable platforms (an approach known as **graceful degradation**).

To exemplify why knowing this in advance matters, consider that if you were unlucky enough to have 25% of your website visitors using Internet Explorer 9 (for example), you'd need to consider what features that browser supports and tailor your solution accordingly. The same caution would be required if large amounts of your users are visiting with older mobile phone platforms such as Android 2. What you can consider a 'base' experience will vary depending upon the project.

If suitable data isn't available, I apply a simple and crude piece of logic to determine whether I should spend time developing a particular platform/browser version: if the cost of developing and supporting browser X is more than the revenue/benefit created by the users on browser X; don't develop specific solutions for browser X.

It's rarely a question of whether you could 'fix' an older platform/version. It's a question of whether you should.

When considering which platforms and browser versions support which features, if you aren't already, become familiar the <http://caniuse.com> website. It provides a simple interface for establishing what browser support there is for the features we will be looking at throughout.



## A brief note on tooling and text editors

It makes no difference what text editor or IDE system you use to build your responsive web designs. If the simplest of text editors allows you to write your HTML, CSS, and JavaScript efficiently, that's absolutely fine. Similarly there are no requisite pieces of tooling that are essential to get a responsive web design out of the door. All you actually need is something that enables you to write HTML, CSS, and JavaScript. Whether your preference is Sublime Text, Vim, Coda, Visual Studio, or Notepad - it matters little. Just use what works best for you.

However, be aware that there are more tools available now (often free) to negate many of the manual and time-intensive tasks of building web sites than ever before. For example, CSS processors (Sass, LESS, Stylus, PostCSS) can help with code organization, variables, color manipulations, and arithmetic. Tools like PostCSS can also automate horrible and thankless jobs like CSS vendor prefixing. Furthermore, 'Linting' and validation tools can check your HTML, JavaScript, and CSS code against standards as you work, eliminating many time wasting typos or syntax errors.

New tools come out constantly and they are continually improving. Therefore, whilst some relevant and beneficial tools will be mentioned by name as we go, be aware that something better may be just around the corner. Hence we won't be relying on anything other than standards based HTML and CSS in our examples. You should however, use whatever tools you can to produce your front-end code as quickly and reliably as possible.

## Our first responsive example

In the first paragraph I promised that by the end of this chapter you would know all you needed to build a fully responsive web page. So far I've just been talking around the issue at hand. It's time to walk the walk.



### Code samples

You can download all the code samples from this book by visiting [rwd.education/download.zip](http://rwd.education/download.zip) or via GitHub at <https://github.com/benfrain/rwd>. It's worth knowing that where individual examples are built up throughout a chapter, only the final version of the example is provided in the code download. For example, if you download the code samples for *Chapter 2, Media Queries – Supporting Differing Viewports*, the examples will be in the state they are at by the end of *Chapter 2, Media Queries – Supporting Differing Viewports*. No intermediate states are provided other than in the text.

## Our basic HTML file

We will start with a simple HTML5 structure. Don't worry at this point what each of the lines do (especially the content of the `<head>`, we will cover that in detail in *Chapter 4, HTML5 for Responsive Web Designs*).

For now, simply concentrate on the elements inside the `<body>` tag. I'm pretty sure nothing there will look too unusual; a few `div`'s, a graphic for a logo, an image (a tasty looking scone), a paragraph or two of text and a list of items.

Here's an abridged version of the code. For brevity I have removed the paragraphs of text in the code below as we only need to concern ourselves with the structure. However, you should know that it's a recipe and description of how to make scones; quintessentially British cakes.

If you want to see the full HTML file, you can download it from the `rwd.education` website.

```
<!doctype html>
<html class="no-js" lang="en">
  <head>
    <meta charset="utf-8">
    <title>Our first responsive web page with HTML5 and CSS3</
title>
    <meta name="description" content="A basic responsive web page
- an example from Chapter 1">
    <link rel="stylesheet" href="css/styles.css">
  </head>
  <body>
    <div class="Header">
      <a href="/" class="LogoWrapper"></a>
      <p class="Strap">Scones: the most resplendent of snacks</
p>
    </div>
    <div class="IntroWrapper">
      <p class="IntroText">Occasionally maligned and
misunderstood; the scone is a quintessentially British classic.</p>
      <div class="MoneyShot">
        
        <p class="ImageCaption">Incredible scones, picture
from Wikipedia</p>
      </div>
    </div>
    <p>Recipe and serving suggestions follow.</p>
    <div class="Ingredients">
```

```

<h3 class="SubHeader">Ingredients</h3>
<ul>

</ul>
</div>
<div class="HowToMake">
  <h3 class="SubHeader">Method</h3>
  <ol class="MethodWrapper">

</ol>
</div>
</body>
</html>

```

By default, web pages are flexible. If you were to open the example page, even as it is at this point (with no media queries present), and resize the browser window you'll see the text reflows as needed.

What about on different devices? With no CSS whatsoever, this is how that renders on an iPhone:



As you can see, it's rendering like a 'normal' web page would on an iPhone. The reason for that is that iOS renders web pages at 980px wide by default and shrinks them down into the viewport.

The viewable area of a browser is known technically as the **viewport**. The viewport is seldom equivalent to the screen size of a device, especially in instances where a user can resize a browser window.

Therefore, from now on, we will generally use this more accurate term when we are referring to the available space for our web page.

We can fix that prior problem easily by adding this snippet in the <head>:

```
<meta name="viewport" content="width=device-width">
```

This viewport meta tag is a non-standard (but de facto standard) way of telling the browser how to render the page. In this case, our viewport meta tag is effectively saying "make the content render at the width of the device". In fact, it's probably easier to just show you the effect this line has on applicable devices:

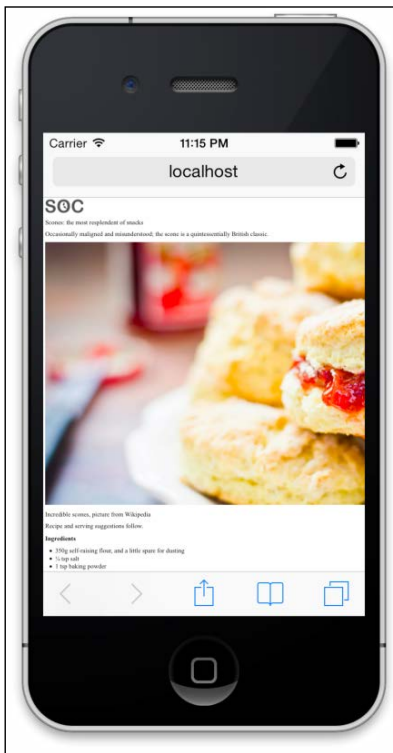


Great! The text is now rendering and flowing at a more 'native' size. Let's move on.

We will cover the `meta` tag and its various settings and permutations (and the standards based version of the same functionality) in *Chapter 2, Media Queries – Supporting Differing Viewports*.

## Taming images

They say a picture is worth a thousand words. All this writing about scones in our sample page and there's no image of the beauties. I'm going to add in an image of a scone near the top of the page; a sort of 'hero' image to entice users to read the page.



Oh! That nice big image (2000px wide) is forcing our page to render more than a little wonky. We need to fix that. We could add a fixed width to the image via CSS but the problem there is that we want the image to scale to different screen sizes.

For example, our example iPhone is 320px wide so we could set a width of 320px to that image but then what happens if a user rotates the screen? The 320px wide viewport is now 480px wide. Thankfully it's pretty easy to achieve fluid images that will scale to the available width of their container with a single line of CSS.