Chapter No. 1
"Responsive Web Design"
In this package, you will find:

The author’s biography
A preview chapter from the book, Chapter no.1 “Responsive Web Design”
A synopsis of the book’s content
Information on where to buy this book

About the Author

**Thoriq Firdaus** is a web developer and lives in Indonesia. He has been dabbling in web design and development for more than 5 years, working with many clients of varying sizes. He appreciates the giving nature of the web design community at large. He also loves trying out new things in CSS3 and HTML5 and occasionally speaks at some local colleges and institutions on the subject.

Outside of work, he loves spending time with his wife and daughter, watching movies, and enjoying meals at nearby cafes and restaurants.

For More Information:
Responsive Web Design by Example
Beginner's Guide
Second Edition

Responsive web design has taken the web design industry by storm. It's not a trend but a norm; it is something that is now normally expected from a website. You might have read and come across many discussions about responsive web design on blogs, forums, Facebook, and Twitter. Likewise, you want your website to be responsive in order to make it presentable on any screen size. Hence, this is the book you are looking for.

This book teaches you how to build presentable, responsive websites through examples, tips, and best practices of code writing and project organization. Additionally, you will also learn how to use CSS preprocessors, LESS, and Sass, which allows you to compose leaner style rules.

What This Book Covers

Chapter 1, Responsive Web Design, looks at the basic principle behind responsive web design, explains the basic technicalities to build a responsive website and features a couple of responsive frameworks as well as the advantages of using one.

Chapter 2, Web Development Tools, helps you to prepare, install, and set up the software to run projects and build responsive websites.

Chapter 3, Constructing a Simple Responsive Blog with Responsive.gs, introduces the Responsive.gs framework and builds the HTML structure of a blog using several HTML5 elements and the Responsive.gs grid system.

Chapter 4, Enhancing the Blog Appearance, composes CSS style rules to enhance the blog’s look and feel. You will also learn how to manage blog styles modularly with partial style sheets and to compile them together into a single style sheet with a compiler.

Chapter 5, Developing a Portfolio Website with Bootstrap, builds a portfolio website with the Bootstrap framework components, including the grid system, button, and form as the base. We also learn how to use Bower to manage project libraries.

Chapter 6, Polishing the Responsive Portfolio Website with LESS, explores and teaches us the use of several LESS features, such as nesting, variables, and mixins, to write leaner and reusable style rules and eventually to polish the responsive portfolio website.

For More Information:
Chapter 7, *A Responsive Website for Business with Foundation*, builds a responsive website for a startup business using the Foundation framework grid system and components.

Chapter 8, *Extending Foundation*, teaches how to use Sass and SCSS syntax, such as variables, mixins, and function, to write maintainable and reusable styles for the responsive startup website.

Appendix, *Pop Quiz Answers*, contains answers to the multiple choice pop quizzes you will find throughout the book.
Responsive Web Design

I still remember, back when I was a kid, a mobile phone came with a mere tiny size monochromatic screen. All we could do at that time was make a phone call, text, and play a simple game. Today, mobile devices have drastically advanced in many ways.

New mobile devices are built with varying screen sizes; some even come with higher DPI or resolution. Most new mobile devices are now equipped with a touch-enabled screen, allowing us to interact with the device conveniently using a tap or a swipe of fingers. The screen orientation is switchable between portrait and landscape. The software is also more capable compared to older devices. The mobile browser, in particular, is now able to render and display web pages that are as good as a browser in a desktop computer.

In addition, the number of mobile users has exploded in the last couple of years. We can now see many people around spending countless hours facing their mobile devices, a phone, or a tablet, doing things such as running their businesses on the go or simple Internet browsing. The number of mobile users is likely to grow in the years to come and may even outnumber the total number of desktop users.

That is to say, mobiles have changed the Web and changed the way people use the Internet and enjoy websites. These advancements in mobile devices and the increasing mobile Internet usage prompts questions on a new paradigm to build websites that are accessible and function well in varying circumstances. This is where Responsive Web Design comes in.

For More Information:
Responsive web design is one of the most discussed topics in the web design and development community. So, I believe many of you have heard about it to a certain extent.

Ethan Marcotte was the one who coined the term "responsive web design". He suggests in his article Responsive Web Design (http://alistapart.com/article/responsive-web-design/), that the Web should seamlessly adjust and adapt to the environment where the users view the website rather than addressing it exclusively for a specific platform. In other words, the website should be responsive, it should be presentable on any screen size, regardless of the platform on which it is viewed.

Take the Time website (http://time.com/) as an example. The web page fits nicely on a desktop browser with a large screen size and also on a mobile browser with a limited viewable area. The layout shifts and adapts as the viewport size changes. As you can see in the following screenshot, on the mobile browser, the background color of the header is dark grey, the image is scaled down proportionally, and the Tap bar appears where Time hides the latest news, magazine, and videos sections:
There are two components to build responsive websites, namely, **viewport meta tag** and **media queries**.

### Viewport meta tag

Before smartphones, such as the iPhone, became mainstream, every website was built to be around 1000 px in width or 980 px wide and it was zoomed out to fit into the mobile phone screen, which eventually made the website unreadable. Hence, the `<meta name="viewport">` was created.

In a nutshell, the viewport meta tag is used to define the web page scale and its visible area (viewport) in the browser. The following code is an example of a viewport meta tag in action:

```html
<meta name="viewport" content="width=device-width, initial-scale=1">
```

The preceding viewport meta tag specification defines the web page viewport width to follow the device. It also defines the web page scale upon opening the web page for the first time at 1:1, in a way that the sizes and the dimensions of the web page content should be persistent; they should not be scaled up or scaled down.

---

For More Information:  
In favor of comprehending how the viewport meta tag would affect a web page layout, I have created two web pages for comparison; one with the viewport meta tag added in and the other one without it. You can see the difference in the following screenshot:

The first website shown in the preceding image is issued with the viewport meta tag using the exact same specification as in our previous code example. As we have specified \texttt{width=device-width}, the browser acknowledges that the website viewport is at the same size as the device screen, so that it will not squish the web page to fit in the entire screen. The \texttt{initial-scale=1} will retain the title and the paragraph in their original size.

In the second website's example, as we did not add the viewport meta tag, the browser assumed that the web page should be displayed entirely. So, the browser forced the whole website down to fit within the whole screen area, making the title and the text totally unreadable.

For More Information:
A word on screen size and viewport

You may have found on many web design forums or blogs that viewport and screen size are mentioned interchangeably quite often. But, as a matter of fact, they are two different things.

Screen size refers to the device’s actual screen size. A 13-inch laptop, for example, commonly has a screen size of 1280*800 pixels. The viewport, on the other hand, describes the viewable area in the browser where it displays websites. The following diagram illustrates this:

![Diagram showing screen size and viewport]

Media queries

The media types module in CSS enables us to target style rules to specific media. If you have created a print style sheet before, you certainly are familiar with the concept of media types. CSS3 introduced a new media type called media queries, which allow us to apply styles within the specified range of the viewport width, also known as breakpoints.

The following is one simple example; we decrease the p font size from 16px to 14px of the website when the website's viewport size is at 480px or lower.

```css
p {
  font-size: 16px;
}
@media screen and (max-width: 480px) {
  p {
    font-size: 14px;
  }
}
```

For More Information:

Responsive Web Design

The following diagram illustrates the preceding code:

We can also combine multiple ranges of viewport widths by using the and operator. Following our preceding example, we can set the p font size to 14px when the viewport size is between 480px and 320px in the following manner:

```css
@media screen and (min-width: 320px) and (max-width: 480px) {
    p {
        font-size: 11px;
    }
}
```

Viewport and media queries references

We will be dealing with viewport meta tag and media queries while building responsive websites. Packt Publishing has published a dedicated book, *Responsive Web Design with HTML5 and CSS3*, Ben Frein, Packt Publishing, which covers these two in greater detail. I would like to suggest reading it as a companion and complement to this book.

A look into responsive frameworks

Building a responsive website can be very tedious work. There are many measurements to be considered while building a responsive website, one of which would be creating the responsive grid.

For More Information:

The grid helps us build websites with proper alignment. If you have ever used 960.gs (http://960.gs/), which is one of the popular CSS frameworks, you would have experienced how easy it is to organize the web page layout by adding preset classes, such as grid_1 or push_1, in the elements.

However, the 960.gs grid is set in a fixed unit, namely, pixel (px), which is not applicable when it comes to building a responsive website. We need a framework with the grid set in percentage (%) unit to build responsive websites; we need a responsive framework.

A responsive framework provides the building blocks to build responsive websites. Generally, it includes the classes to assemble a responsive grid, the basic styles for typography and form inputs, and a few styles to address various browser quirks. Some frameworks even go further with a series of styles to create common design patterns and web user interfaces such as buttons, navigation bars, and image slider. These predefined styles allow us to develop responsive websites faster with less hassle. The following are a few other reasons why using a responsive framework is a favorable option to build responsive websites:

- **Browser compatibility**: Assuring the consistency of a web page on different browsers is really painful and more distressing than developing the website itself. However, with a framework, we can minimize the work to address browser compatibility issues. The framework developers have most likely tested the framework on various desktop browsers and mobile browsers with the most constrained environment prior to releasing it publicly.

- **Documentation**: A framework, in general, also comes with comprehensive documentation that records the bits and pieces on using the framework. The documentation will be very helpful for entry users to begin studying the framework. It is also a great advantage when we are working with a team. We can refer to the documentation to get everyone on the same page and follow the standard code of writing conventions.

- **Community and extensions**: Some popular frameworks such as Bootstrap and Foundation have an active community that helps address the bugs in the framework and extends the functionality. The jQuery UI Bootstrap (http://jquery-ui-bootstrap.github.io/jquery-ui-bootstrap/) is perhaps a good example in this case. The jQuery UI Bootstrap is a collection styles for jQuery UI widgets to match the feel and look of Bootstrap's original theme. It’s now common to find free WordPress and Joomla themes that are based on these frameworks.

Through the course of this book, we will be building three responsive websites by using three different responsive frameworks, namely Responsive.gs, Bootstrap, and Foundation.
Responsive Web Design

The Responsive.gs framework

Responsive.gs (http://responsive.gs/) is a lightweight, responsive framework, which is merely 1 KB in size when compressed. Responsive.gs is based on a width of 940 px, and made in three variants of grids, that is, 12, 16, and 24 columns. What’s more, Responsive.gs is shipped with box-sizing polyfill, which enables CSS3 box-sizing in Internet Explorer 6, 7, and 8, and makes it decently presentable in those browsers.

Polyfill is a piece of code that enables certain web features and capabilities that are not built in the browser natively. Usually, it addresses the older versions of Internet Explorer. For example, you can use HTML5Shiv (https://github.com/aFarkas/html5shiv) so that new HTML5 elements, such as <header>, <footer>, and <nav>, are recognized in Internet Explorer 6, 7, and 8.

A word on CSS box model

HTML elements, which are categorized as block-level elements, are essentially boxes drawn with the content width, height, margin, padding, and border through CSS. Prior to CSS3, we were facing constraints when specifying a box. For instance, when we specify a <div> tag with a width and height of 100px, as follows:

```css
div {
    width: 100px;
    height: 100px;
}
```

The browser will render div as a 100px square box, as shown in the next diagram:
However, this will only be true if the padding and border have not been added in. As a box has four sides, a padding of 10px (`padding: 10px;`) will actually add 20px to the width and height—10px for each side, as shown in the following diagram:

While it takes up space on the page, the element’s margin space is reserved outside the element rather than as part of the element itself; thus, if we give an element a background color, the margin area will not take that color.

**CSS3 box sizing**

CSS3 introduced a new property called `box-sizing`, which lets us specify how the browser should calculate the CSS box model. There are a couple of values that we can apply within the `box-sizing` property.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>content-box</code></td>
<td>This is the default value of the box model. This value specifies the padding and the border-box’s thickness outside the specified width and height of the content, as we have demonstrated in the preceding section.</td>
</tr>
<tr>
<td><code>border-box</code></td>
<td>This value will do the opposite of what the content-box does; it includes the padding and the border box as the width and height of the box.</td>
</tr>
<tr>
<td><code>padding-box</code></td>
<td>At the time of writing this book, this value is experimental and has just been added recently. This value specifies the box dimensions.</td>
</tr>
</tbody>
</table>

For More Information:

Responsive Web Design

In each of the projects in this book, we will be using the `border-box` value so that we can determine the box dimensions with ease for the websites. Let’s take our preceding example to understand this, but this time we will set the `box-sizing` model to `border-box`. As mentioned in the preceding table, the `border-box` value will retain the box’s width and the height as 100px, regardless of the padding and border addition. The following illustration shows a comparison between the outputs of the two different values, `content-box` (the default value) and `border-box`:

In this book, we will use Responsive.gs and explore more of it in the next two chapters to build a simple responsive blog.

For More Information:
The Bootstrap framework

Bootstrap (http://getbootstrap.com/) was originally built by Mark Otto (http://markdotto.com/) and initially intended only for internal use on Twitter. In short, Bootstrap was then launched for free for public consumption.

Bootstrap has long been associated with Twitter, but since the author has departed from Twitter and Bootstrap itself has grown beyond his expectations, Bootstrap now stands on its own brand (http://blog.getbootstrap.com/2012/09/29/onward/).

If you refer to the initial development, the responsive feature was not yet added. It was then added in Version 2 due to the increasing demand for creating responsive websites.

Bootstrap also comes with many more added features as compared to Responsive.gs. It is packed with preset user interface styles, which comprise common user interfaces used on websites such as buttons, navigation bars, pagination, and forms, so you don’t have to create them from scratch again when starting a new project. On top of that, Bootstrap is also powered with some custom jQuery plugins such as image slider, carousel, popover, and modal box.

You can use and customize Bootstrap in many ways. You can directly customize the Bootstrap theme and its components directly through the CSS style sheets, the Bootstrap customize and download page (http://getbootstrap.com/customize/), or the Bootstrap LESS variables and mixins, which are used to generate the style sheets.

In this book, we will go into Bootstrap in Chapter 5, Developing a Portfolio Website with Bootstrap, and Chapter 6, Polishing the Responsive Portfolio Website with LESS, to build a responsive portfolio website.

The Foundation framework

Foundation (http://foundation.zurb.com/) is a framework created by ZURB, a design agency based in California. Similar to Bootstrap, Foundation is not just a responsive CSS framework; it is shipped with a preset grid, components, and a number of jQuery plugins to present interactive features.

Some high-profile brands, such as McAfee (http://www.mcafee.com/common/privacy/english/slide.html), which is one of the most respectable brands for computer antivirus, have built their websites using Foundation.

For More Information:
Responsive Web Design

The Foundation style sheet is powered by Sass, a Ruby-based CSS preprocessor. We will be discussing more about Sass, along with the Foundation features in the last two chapters of this book; therein, we will be developing a responsive website for a startup company.

There are many complaints that the code in responsive frameworks is excessive; as a framework such as Bootstrap is used widely, it has to cover every design scenario and thus, it comes with some extra styles that you might not need for your website. Fortunately, we can easily minimize this issue by using the right tools, such as CSS preprocessors, and following a proper workflow.

Frankly, there isn’t a perfect solution; and using a framework certainly isn’t for everyone. It all comes down to your needs, your website’s needs, and in particular, your client’s needs and budgets. In reality, you will have to weigh these factors to decide whether you will go with a responsive framework or not. Jem Kremer has an extensive discussion in this regard in her article Responsive Design Frameworks: Just Because You Can, Should You? (http://www.smashingmagazine.com/2014/02/19/responsive-design-frameworks-just-because-you-can-should-you/)

A brief introduction to CSS preprocessors

Both Bootstrap and Foundation use CSS preprocessors to generate their style sheets. Bootstrap uses LESS (http://lesscss.org/)—though the official support for Sass has just been released recently. Foundation, on the contrary, uses Sass as the only way to generate its style sheets (http://sass-lang.com/).

CSS preprocessor is not an entirely new language. If you have known CSS, you should be accustomed to the CSS preprocessor immediately. The CSS preprocessor simply extends CSS by allowing the use of programming features such as variables, functions, and operations.

The following is an example of how we write CSS with the LESS syntax:

```css
@color: #f3f3f3;
body {
    background-color: @color;
}
p {
    color: darken(@color, 50%);
}
```

For More Information:

When the preceding code is compiled, it takes the @color variable that we have defined and places the value in the output, as follows:

```css
body {
    background-color: #f3f3f3;
}
p {
    color: #737373;
}
```

The variable is reusable throughout the style sheet and this enables us to retain style consistency and make the style sheet more maintainable.

We are going to use and explore CSS preprocessors, LESS, and Sass further during the course of building responsive websites with Bootstrap (Chapter 5, Developing a Portfolio Website with Bootstrap and Chapter 6, Polishing the Portfolio Website with LESS) and Foundation (Chapter 7, A Responsive Website for Business with Foundation, and Chapter 8, Extending Foundation).

### Have a Go Hero — delve into responsive web design

Our discussion on responsive web design here, though essential, is merely the tip of the iceberg. There is so much more about responsive web design than what we have recently covered in the preceding sections. I would suggest that you take your time to get yourself more insight and remove any apprehension on responsive web design, including the concept, the technicalities, and some constraints.

The following are some of the best recommendations for references:

- The [Responsive Web Design](http://alistapart.com/article/responsive-web-design) article by Ethan Marcotte, is where it all begins
- Also a good place to start is [Responsive Web Design](http://webdesign.tutsplus.com/articles/responsive-web-design--) by Rachel Shillcock
- [Don’t Forget the Viewport Meta Tag](http://webdesign.tutsplus.com/articles/quick-tip-dont-forget-the-viewport-meta-tag--) by Ian Yates
- [How To Use CSS3 Media Queries To Create a Mobile Version of Your Website](http://www.smashingmagazine.com/2010/07/19/how-to-use-css3-media-queries-to-create-a-mobile-version-of-your-website/) by Rachel Andrew

For More Information:

Responsive Web Design

- Read about the future standards on responsive image using HTML5 Picture Element. 
  *Responsive Images Done Right: A Guide To <picture> And srcset* by Eric Portis 
  [http://www.smashingmagazine.com/2014/05/14/responsive-images-done-right-guide-picture-srcset/]

- A roundup of methods to make data table responsive. 

### Pop Quiz — responsive web design main components

Q1. In his article, which we have referred to about two times in this chapter, Ethan Marcotte mentioned the main technical ingredients that formulate a responsive website. What are those main components?

1. Viewport Meta Tag and CSS3 Media Queries.
2. Fluid grids, flexible images, and media queries.
3. Responsive images, breakpoints, and polyfills.

Q2. What is a viewport?

1. The screen size of the device.
2. The region where the web page is rendered.
3. The meta tag to set the web page's viewport size.

Q3. Which one of these is the correct way to declare CSS3 Media Queries?

1. ```
@media (max-width: 320px) { p{ font-size:11px; }}
```  
2. ```
@media screen and (max-device-ratio: 320px) { div{ color:white; }}
```  
3. ```
<link rel="stylesheet" media="(max-width: 320px)" href="core.css" />
```
Responsive web design inspiration sources

Now, before we jump into the next chapters and start building responsive websites, it may be a good idea to spend some time looking for ideas and inspiration for responsive websites; to see how they are built and how the layout is organized on desktop browsers, as well as on mobile browsers.

It’s a common thing for websites to be redesigned from time to time to stay fresh. So, instead of making a pile of website screenshots, which may no longer be relevant in the next several months because of the redesign, we’re better going straight to the websites that curate websites, and the following are the places to go:

- MediaQueries (http://mediaqueri.es/)
- Awwwards (http://www.awwwards.com/websites/responsive-design/)
- CSS Awards (http://www.cssawards.net/structure/responsive/)
- WebDesignServed (http://www.webdesignserved.com/)
- Bootstrap Expo (http://expo.getbootstrap.com/)
- Zurb Responsive (http://zurb.com/responsive)

Summary

In this chapter, we glanced at the short story behind responsive web design, as well as the viewport meta tag and CSS3 media queries, which formulate responsive websites. This chapter also concluded that we are going to work on three projects by using the following frameworks: Responsive.gs, Bootstrap, and Foundation.

Using a framework is an easier and faster way to get responsive websites up and running, rather than building everything from scratch on our own. Alas, as mentioned, using a framework also has some negative aspects. If it is not done properly, the end result could all go wrong. The website could be stuffed and stuck with unnecessary styles and JavaScript, which at the end makes the website load slowly and hard to maintain.

We need to set up the right tools; not only will they facilitate the projects, but they will also help us in making the website more easy to maintain, and this is what we are going to do in the next chapter.

For More Information:
Where to buy this book


Free shipping to the US, UK, Europe and selected Asian countries. For more information, please read our shipping policy.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.

www.PacktPub.com