



C o m m u n i t y E x p e r i e n c e D i s t i l l e d

Magento Responsive Theme Design

Leverage the power of Magento to successfully develop and deploy a responsive Magento theme

Richard Carter

[PACKT] open source*
PUBLISHING community experience distilled

Magento Responsive Theme Design

Leverage the power of Magento to successfully develop and deploy a responsive Magento theme

Richard Carter



BIRMINGHAM - MUMBAI

Magento Responsive Theme Design

Copyright © 2013 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: December 2013

Production Reference: 1171213

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham B3 2PB, UK.

ISBN 978-1-78398-036-9

www.packtpub.com

Cover Image by Jarek Blaminsky (milak6@wp.pl)

Credits

Author

Richard Carter

Project Coordinator

Ankita Goenka

Reviewers

Oğuz Çelikdemir

Vinai Kopp

Proofreader

Lindsey Thomas

Acquisition Editor

Sam Wood

Indexer

Priya Subramani

Commissioning Editor

Mohammed Fahad

Production Coordinator

Arvindkumar Gupta

Technical Editors

Neha Mankare

Siddhi Rane

Cover Work

Arvindkumar Gupta

About the Author

Richard Carter is a web designer and frontend web developer based in Newcastle upon Tyne in the North East of England.

His experience includes many open source e-commerce and content management systems, including Magento, MediaWiki, WordPress, and Drupal. He has worked with clients including the University of Edinburgh, University College Dublin, Directgov, NHS Choices, and BusinessLink.gov.uk.

He is a creative director at Peacock Carter Ltd (peacockcarter.co.uk), a web design and development agency based in the North East of England. He graduated from the University of Durham in Software Engineering, and currently lives in Newcastle upon Tyne. He blogs at earlgreyandbattenburg.co.uk and tweets as @RichardCarter and @PeacockCarter.

This is his sixth book. He has previously written *MediaWiki Skins Design*, *Magento 1.3 Theme Design*, *Magento 1.4 Theme Design*, *Joomla! 1.5 Templates Cookbook*, and *The Beginner's Guide to Drupal Commerce*, and acted as a technical reviewer on *MediaWiki 1.1 Beginners Guide*, *Inkscape 0.48 Illustrator's Cookbook*, and *Apress' The Definitive Guide To Drupal 7*.

In particular, my thanks are due to Matthew, who has kept Peacock Carter on track while I've focused on the book! Thanks also to my family and friends, whose constant support is much appreciated.

About the Reviewers

Oğuz Çelikdemir is a senior software developer specializing in web application development who works for Metro Group (www.metrogroup.de), an affiliated company in Turkey.

He has been working in the IT sector since 1990 and he has implemented many e-commerce websites using Magento. He is also an experienced PHP and ExtJS framework user.

He lives in Istanbul with his wife, Neşe, and his twins, Ece and Efe.

Vinai Kopp has worked as an independent web developer since 1998, learning the nuts and bolts of protocols, databases, languages, and standards as the modern Internet evolved. In March 2008, he specialized in development for the Magento platform, and has since collected experience in many projects, sized from small shops to large-scale projects in highly integrated implementations.

During 2012 and 2013, he worked as the senior manager of Developer Education at Magento Inc., creating and delivering technical training for Magento developers around the world.

Now he works as an independent consultant, developer, and trainer.

He has also played a part in the creation of the certifications for Magento developers and frontend engineers, and is a co-author of the *German Magento Developer Handbook*. At numerous Magento community events and at Magento Imagine, he has presented as a subject matter expert on technical topics.

In his spare time, he develops and maintains popular free and open source Magento modules, which can be found on GitHub at <http://github.com/Vinai/>.

He also loves running. At Magento events, you will often find him geeking out and going for early morning runs with likeminded friends.

Besides having fun with development in general and Magento in particular, he has a great wife and two lovely daughters.

www.PacktPub.com

Support files, eBooks, discount offers, and more

You might want to visit www.PacktPub.com for the support files and downloads related to your book.

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 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 for more details.

At 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.



<http://PacktLib.PacktPub.com>

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

Free Access for Packt account holders

If you have an account with Packt at www.PacktPub.com, you can use this to access PacktLib today and view nine entirely free books. Simply use your login credentials for immediate access.

Table of Contents

Preface	1
Chapter 1: Beginning a Responsive Magento Theme	5
Creating your Magento theme	5
Adding the basic CSS styling for your Magento theme	6
Adding the basic XML layout for your Magento theme	7
Adding the meta viewport element	8
A note on Magento theme hierarchy	9
Magento theme hierarchy	10
Enabling your new theme in Magento	11
Overwriting the default Magento templates	14
Adding the media queries to your Magento theme	15
Some other common breakpoints	19
Styling images responsively in your Magento theme	19
Styling e-commerce navigation responsively in Magento	24
Adding skip to footer navigation to your Magento store	25
Adding the skip-to link in your header template	26
Adding the footer sitemap navigation	29
Dropdown navigation for your Magento store	31
Summary	36
Chapter 2: Making Your Store Responsive	37
Laying out your website's header and footer	37
Adding the header and footer CSS	38
Adding CSS for larger screens	39
Adding CSS to change header and footer links	40
Responsive product page layout in Magento	42
Laying out the product image and product information	43
Making a definitive style	44
Responsive category page layout	46
Category list view	46

Product pager, pagination, and sort by dropdown	48
Category grid view	50
Dealing with Magento search results responsively	52
Resizing product images	53
Removing the height and width attributes	54
Summary	56
Chapter 3: Responsive Checkout and Cart in Magento	57
Responsive shopping cart in Magento	57
Styling form elements	58
Removing the shipping estimate tool from the Magento cart page	62
Styling cart tables	62
Removing unnecessary table columns for mobile and smaller screened devices	64
Responsive one page checkout in Magento	68
Styling Magento customer account pages responsively	73
Setting the account login page template to a one-column layout	73
Styling error messages	76
Removing the My Applications and My Downloadable Products links from the Magento customer account area	78
Summary	80
Chapter 4: Enhancing Your Responsive Magento Theme	81
Supporting CSS media queries in Internet Explorer with css3-mediaqueries.js	81
Adding a JavaScript file to your Magento theme through local.xml	83
Improving Magento store data entry for customers	84
Changing your Magento theme's doctype to HTML5	84
Changing the input type value for login	85
Changing the input type value for registration	86
Using the CSS @font-face rule to use custom fonts in your Magento theme	89
Summary	91
Index	93

Preface

E-commerce has changed drastically in the last few years, with the need for stores to provide a realistic interface for both mobile and desktop users becoming increasingly important.

Responsive design is one approach to this requirement, and this book begins to help you uncover how you can apply responsive web design techniques within Magento.

What this book covers

Chapter 1, Beginning a Responsive Magento Theme, begins building your new responsive Magento theme and installing it on your Magento store.

Chapter 2, Making Your Store Responsive, adds to what you started in the previous chapter, dealing with Magento-specific pages from product pages to category listings.

Chapter 3, Responsive Checkout and Cart in Magento, looks at the Magento cart and checkout in more detail to address key issues customers are likely to have during the order process.

Chapter 4, Enhancing Your Responsive Magento Theme, adds some improvements to your new theme to improve your customers' experiences.

What you need for this book

You will need access to a recent copy of the Magento Community Edition (1.7 or newer).

Who this book is for

This book is for web designers and developers with an existing knowledge of modern HTML and CSS. Knowledge of Magento theming or responsive web design techniques isn't required though may be advantageous.

Conventions

In this book, you will find a number of styles of text 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: "Using the CSS `@font-face` rule to use custom fonts in your Magento theme."

A block of code is set as follows:


```
.breadcrumbs li {
    color: #777;
    display: inline;
}
.breadcrumbs a {
    color: #777;
}


.quick-access {
    color: #777;
    text-align: right;
}
```

When we wish to draw your attention to a particular part of a code block, the relevant lines or items are set in bold:

```
.quick-access .links,
.footer ul {
    list-style-type: none;
}
.quick-access .links li,
.footer ul li {
    display: inline;
}
```

New **terms** and **important words** are shown in bold. Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: "along with the **My Account** link and other related links."

[ 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 may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply send an e-mail to feedback@packtpub.com, and mention the book title via 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 on 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 for all Packt books you have purchased from your account at <http://www.packtpub.com>. 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.

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 would 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 on our website, or added to any list of existing errata, under the Errata section of that title. Any existing errata can be viewed by selecting your title from <http://www.packtpub.com/support>.

Piracy

Piracy of copyright 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 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

You can contact us at questions@packtpub.com if you are having a problem with any aspect of the book, and we will do our best to address it.

1

Beginning a Responsive Magento Theme

Every theme starts somewhere, and your responsive Magento theme is not any different. In this chapter you will cover:

- Creating a basic Magento theme
- Enabling your new theme in Magento
- Adding media queries to your theme to establish breakpoints for different device widths
- Styling static content pages in your Magento store
- The various options available to you to provide navigation for both desktop and mobile/tablet users of your store

Creating your Magento theme

Before you can begin building your Magento theme, you will need to create the directory structure that Magento requires its themes to follow. To do this, you will need to create these directories in your Magento installation directory:

- `app/design/frontend/default/responsive`
- `app/design/frontend/default/responsive/template`
- `app/design/frontend/default/responsive/layout`
- `skin/frontend/default/responsive`
- `skin/frontend/default/responsive/images`
- `skin/frontend/default/responsive/css`

This means that we've created a Magento theme called `responsive` in the default directory. Magento themes are split into a number of components. Theme files that are directly included by the browser and processed there, are located in the `skin/` directory of your Magento theme.

Theme files such as `.phtml` templates and layout XML files, which first need to be processed by Magento before they are sent to the browser, are located in the `app/design/` directory of your Magento installation.

No files in the `app/` directory are directly accessible by browsers. If you try to access files in this directory directly, you will receive a forbidden response. The `app/design` directory will contain Magento's `.phtml` templates that define the HTML that is output in to the page. The `skin` directory will contain the CSS and images the design requires to function.

Finally, you will need to create two files to start your new Magento theme:

1. A CSS file called `styles.css` in the `skin/frontend/default/responsive/css` directory; this is where you will start adding to the CSS for your responsive Magento theme.
2. An XML layout file called `local.xml` in the `app/design/frontend/default/responsive/layout` directory.

Adding the basic CSS styling for your Magento theme


In the `styles.css` file you created in the preceding section, you can start by defining some basic styles to reset browser defaults:

```
/* http://meyerweb.com/eric/tools/css/reset/
   v2.0 | 20110126
   License: none (public domain)
*/
html, body, div, span, applet, object, iframe, h1, h2, h3, h4, h5,
h6, p, blockquote, pre, a, abbr, acronym, address, big, cite,
code, del, dfn, em, img, ins, kbd, q, s, samp, small, strike,
strong, sub, sup, tt, var, b, u, i, center, dl, dt, dd, ol, ul,
li, fieldset, form, label, legend, table, caption, tbody, tfoot,
thead, tr, th, td, article, aside, canvas, details, embed,
figure, figcaption, footer, header, hgroup, menu, nav, output,
ruby, section, summary, time, mark, audio, video {margin:
0;padding: 0;border: 0;font-size: 100%;font: inherit;
vertical-align: baseline;}
/* HTML5 display-role reset for older browsers */
```

```

article, aside, details, figcaption, figure, footer, header,
hgroup, menu, nav, section {display: block;}
body {line-height: 1;}
ol, ul {list-style: none;}
blockquote, q {quotes: none;}
blockquote:before, blockquote:after, q:before, q:after {content:
    '';content: none;}
table {border-collapse: collapse;border-spacing: 0;}

```

 Alternatively, you could use `normalize.css` in place of this, which is available on Github at <http://necolas.github.io/normalize.css/>.

Next, you can add some style to begin to help your Magento store take shape:

```

body
{
    background: #EFEFEF;
    font: normal 80%/150% Arial, Helvetica, sans-serif;
}
.page
{
    background: #fff;
    border-radius: 10px;
    margin: 10px;
    padding: 10px;
}
.footer-container
{
    clear: both;
}

```

 You can change your Magento store's logo by navigating to **System | Configuration | Design | Header** in the administration panel.

Adding the basic XML layout for your Magento theme

In the `local.xml` file, add the following code:

```

<?xml version="1.0"?>
<layout version="0.1.0">

```

```
<default>
<reference name="root">
    <action method="setTemplate">
        <template>page/2columns-left.phtml</template>
    </action>
    <action method="setIsHandle">
        <applied>1</applied>
    </action>
</reference>

<remove name="checkout_cart_link"/>
<remove name="right.permanent.callout"/>
<remove name="right.poll"/>

<remove name="paypal.partner.right.logo"/><remove name="cart_
sidebar"/>
<remove name="left.permanent.callout"/>

</default>
</layout>
```

Downloading the example code



You can download the example code files for all Packt books you have purchased from your account at <http://www.packtpub.com>. 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.

The preceding code tells Magento to remove some blocks from all the pages in your store that you are unlikely to use, such as the cart block for the sidebar, and the advertising "callouts".

Adding the meta viewport element

So far, so good, but there is one last thing you'll need to take care of to get your responsive theme functioning on mobile devices—the meta viewport element:

```
<meta name="viewport" content="width=device-width,
    initial-scale=1.0" />
```

This element is placed in the `<head>` of your Magento theme, and tells browsers to scale the width of the canvas that the website is drawn on, to be the width of the device.

Without this, mobile browsers will ignore the breakpoints you added to your Magento theme's CSS file and will attempt to draw your website at its full (desktop) width. Breakpoints are used in your theme's CSS file(s), and tell browsers to only apply the given CSS within the media query to browsers that match the screen width (or height) specified in the media query itself.

To add this element to your Magento theme, open the `local.xml` file in the `app/design/frontend/base/default/layout/` directory and add the following lines:

```
<default>
  <reference name="head">
    <block type="core/text" name="meta.viewport">
      <action method="setText">
        <meta><![CDATA[<meta name="viewport" content="width=device-width, initial-scale=1.0" />]]></meta>
      </action>
    </block>
  </reference>
</default>
```

Save the changes to your file in the `app/design/frontend/default/responsive/layout/` directory, and you're done!

A note on Magento theme hierarchy

Magento has a clever theme hierarchy system which allows a theme to only contain the files it wants to overwrite from the fallback themes. If you don't include a file in your new Magento theme, Magento looks at the fallback directories for the file instead.



Never make changes to the files in the fallback directories when you're creating a Magento theme. The next upgrade to Magento is likely to overwrite any changes made in the fallback directory!

This system makes upgrading Magento less intensive if you have a well written theme, which only overwrites the specific elements of Magento pages that you want to change.



For more information on how Magento's theme hierarchy and inheritance system works, see <http://www.magentocommerce.com/knowledge-base/entry/magentos-theme-hierarchy>.

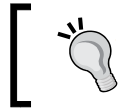
Magento theme hierarchy

As mentioned, Magento uses a clever fallback system for theme files, meaning you only have to overwrite the files you want to change from Magento's defaults. The hierarchy of what Magento checks and in what order is as follows:

1. Check the configured file type specific theme in the configured package.
2. Check the configured default theme in the configured package.
3. Check the literal default theme in the configured package.
4. Check the literal default theme in the base package.

Fallback level 1 can be configured by specifying a theme setting for Templates, Translations, Skin Files, and Layout files in the **System | Configuration | Design** screen. Fallback level 2 can be configured by specifying a theme setting in the field labeled as **Default**. In level 3 only the package is configurable, the theme is always called `default` (in lower case). Fallback level 4 is not configurable.

If a design change is configured for a store view under the **System | Design** screen, that theme will replace the level 1 setting for that store view.



For more information on Magento theme hierarchy and fallbacks, see <http://www.magentocommerce.com/knowledge-base/entry/magentos-theme-hierarchy>.

Enabling your new theme in Magento

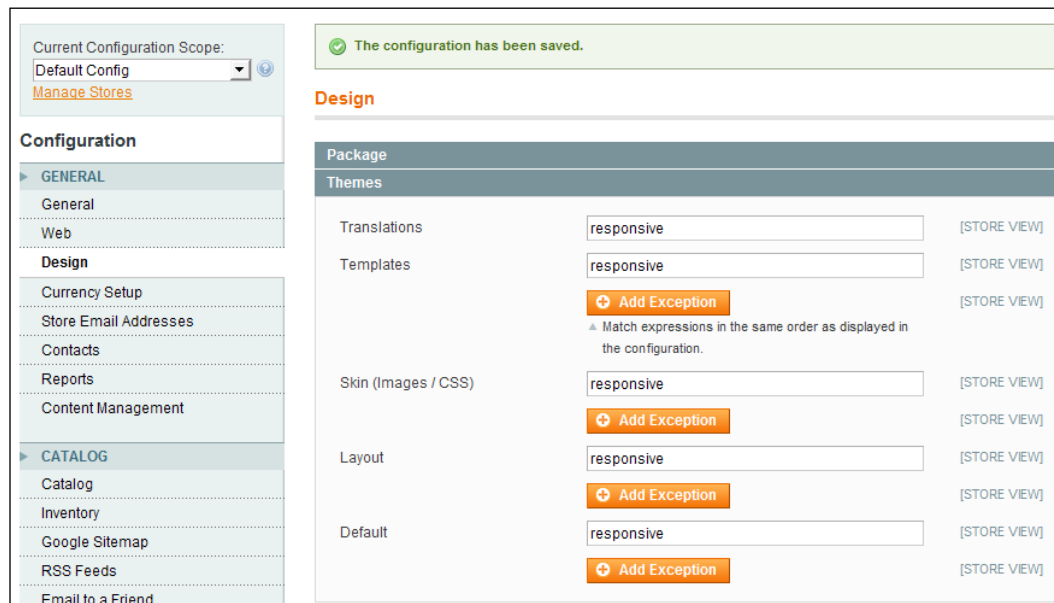
Now that your new theme is in place, you can enable it in Magento. Log in to your Magento store's administration panel. Once you have logged in, navigate to **System | Configuration**, as shown in the following screenshot:

The screenshot shows the Magento Admin Panel interface. At the top, there's a navigation bar with the Magento logo, 'Admin Panel', a search bar, and user information. Below this is a main menu with tabs like Dashboard, Sales, Catalog, Mobile, Customers, Promotions, Newsletter, CMS, Reports, System, and a help icon. The 'System' tab is selected, and its dropdown menu is open, showing options like My Account, Notifications, Tools, Web Services, Design, Import/Export, Manage Currency, Transactional Emails, Custom Variables, Permissions, Magento Connect, Cache Management, Index Management, Manage Stores, Order Statuses, and Configuration. The 'Configuration' option is highlighted with a red box. The main content area shows the 'Dashboard' with various widgets for Lifetime Sales, Average Orders, Last 5 Orders, Last 5 Search Terms, and Top 5 Search Terms, all showing 'No records found'.

From there, select the global configuration scope (labeled **Default Config** in the following screenshot) you want to apply your new theme to, from the **Current Configuration Scope** dropdown in the top left of your screen:



Once this has loaded, navigate to the **Design** tab under **GENERAL** in the left-hand column and expand the **Themes** block in the right-hand column, as shown in the following screenshot:



From here, you can tell Magento to use your new theme. The values given here correspond to the name you gave to the directories when creating your theme. The example uses `responsive` as the value here, as shown in the following screenshot:

The screenshot shows the 'Design' configuration page in Magento. At the top right, there is a 'Save Config' button. The page is divided into two main sections: 'Package' and 'Themes'. Under 'Themes', there are several configuration options, each with a text input field, an 'Add Exception' button, and a 'Use Website' checkbox. The input fields for 'Templates', 'Skin (Images / CSS)', 'Layout', and 'Default' are all set to 'responsive' and are highlighted with red boxes. The 'Translations' field is empty. The 'Use Website' checkboxes are checked for 'Translations', 'Skin (Images / CSS)', 'Layout', and 'Default', and unchecked for 'Templates'.

Section	Field	Value	Use Website
Translations	Translations		<input checked="" type="checkbox"/>
	Use Website		[STORE VIEW]
Templates	Templates	responsive	<input type="checkbox"/>
	Use Website		[STORE VIEW]
Skin (Images / CSS)	Skin (Images / CSS)	responsive	<input checked="" type="checkbox"/>
	Use Website		[STORE VIEW]
Layout	Layout	responsive	<input checked="" type="checkbox"/>
	Use Website		[STORE VIEW]
Default	Default	responsive	<input checked="" type="checkbox"/>
	Use Website		[STORE VIEW]

Click on the **Save Config** button at the top of your screen to save the changes.

Next, check that your new theme has been activated. Remember the `styles.css` file you added in the `skin/frontend/default/responsive/css` directory? The presence of that file is telling Magento to load your new theme's CSS file instead of the default `styles.css` file for Magento from the default package, so your store now has none of the original CSS styling it. As such, you should see the following screenshot when you attempt to view the frontend of your Magento store:



Overwriting the default Magento templates

Noticed the name of your Magento theme appearing next to the logo in the header of your store? You can overwrite the default `header.phtml` that's causing it by copying the contents of `app/design/frontend/base/default/template/page/html/header.phtml` into `app/design/frontend/default/responsive/template/page/html/header.phtml`. Open the file and find the following lines:

```
<?php if ($this->getIsHomePage()):?>
<h1 class="logo"><strong><?php echo $this->getLogoAlt()
?></strong><a href="<?php echo $this->getUrl('') ?>" title=
"<?php echo $this->getLogoAlt() ?>" class="logo"><img src=
```

```


"=php echo $this-&gt;getLogoSrc() ?" alt="=php echo
$this-&gt;getLogoAlt() ?" /></a></h1>
<?php else:?>
<a href="=php echo $this-&gt;getUrl('') ?" title="=php echo
$this-&gt;getLogoAlt() ?" class="logo"><strong>=php echo
$this-&gt;getLogoAlt() ?&gt;&lt;/strong&gt;&lt;img src="<?=php echo
$this-&gt;getLogoSrc() ?" alt="=php echo $this-&gt;getLogoAlt()
?&gt;" /&gt;&lt;/a&gt;
&lt;?php endif?&gt;
</pre

```

Replace them with these lines:

```

<a href="=php echo $this-&gt;getUrl('') ?" title="=php echo $this-
&gt;getLogoAlt() ?" class="logo">` element red if the viewport (space available within the browser window) has a minimum width of 50 em.

[  For a good read on why pixels and other absolute units are not a good idea in media queries, see <http://blog.cloudfour.com/the-ems-have-it-proportional-media-queries-ftw/>. ]

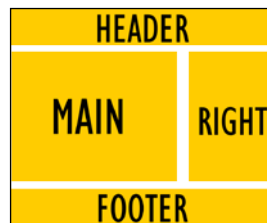
Using this media query, we can start applying widths and floats to the key column elements in Magento to provide a more traditional two-column or three-column layout for desktop and larger tablet browsers:

```
@media only screen and (min-width: 50em)
{
 .col-main, .col-left, .col-right, .col-wrapper
 {
 display: inline;
 margin: 1%;
 padding: 2%;
 }
 .col-left,
 .col-right,
 .col-wrapper,
 .col2-right-layout .col-main
 {
 float: left
 }
 .col-wrapper,
 .col-main
 {
 width: 69%
 }
 .col-left,
 .col-right
 {
 width: 19%
 }
}
```

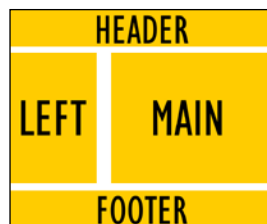
```
}
.col-main
{
 float: right
}

/* Alter column widths for 3 column layout */
.col3-layout .col-main
{
 width: 60%
}
.col3-layout .col-left
{
 width: 27%
}
}
```

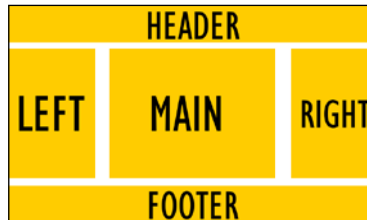
Add this CSS to your theme's `styles.css` file. By floating and setting widths on the columns within the media query, you are telling browsers with a large enough screen to present the columns for a two-column layout with a right-hand column, as shown in the following screenshot:



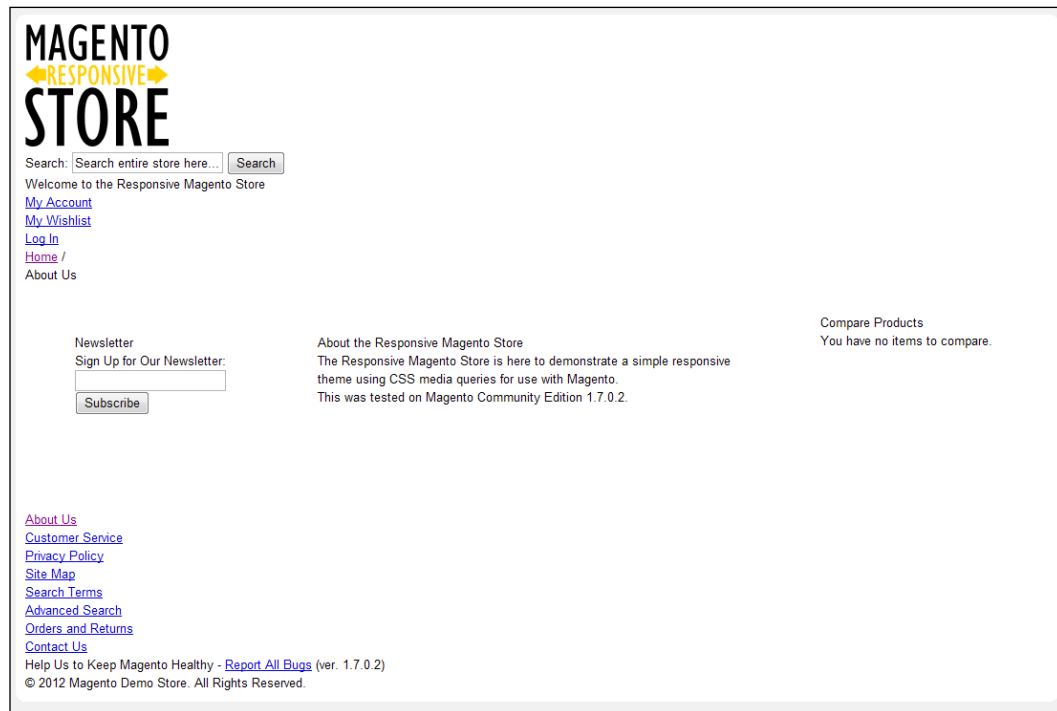
Similarly, for a two-column layout with a left-hand column, you are telling the browser to display the columns as shown in the following screenshot:



Finally, there is a three-column layout with both a left-hand and right-hand column, as shown in the following screenshot:



If you now save those changes and review the frontend of your Magento store, you should see that, on larger-width screens, these layouts are now working again (the following screenshot is set to a three-column layout):



That's it! The very basics of your responsive theme's media queries are in place. You can add more media queries to target more specific devices as you wish. You may want to add one for smaller tablet devices as an intermediate breakpoint between smartphones and desktop computers.

## Some other common breakpoints

So, you have used a breakpoint for desktop devices, but what about if you want to target screen sizes of smaller tablet screens? You can use a breakpoint such as the following code snippet:

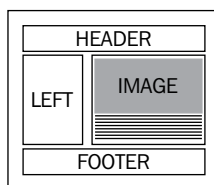
```
@media only screen and (min-width: 35em)
{
 /* Add additional CSS here to target smaller tablets */
}
```

You can also target very large screens (which may include televisions) with the following code snippet:

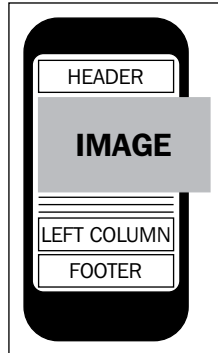
```
@media only screen and (min-width: 65em)
{
 /* Add additional CSS here to target much larger screens*/
}
```

## Styling images responsively in your Magento theme

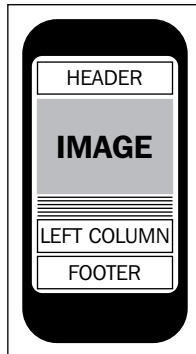
One of the most obvious challenges to address in a responsive website is how to handle images in your content. Consider the following screenshot, as viewed on a desktop computer:



The following screenshot shows what happens if your very-wide image is now viewed on a smaller screen, such as a smartphone screen:




Without some CSS to resize the image to the available space, the image will be displayed at its native size. So if the image was 800 x 500 pixels in size and the screen available is 320 pixels wide, over half of the image would not be displayed by default. Ideally, you want the image to fill the width available so that the entire image is visible to your store's customers, as shown in the following screenshot:



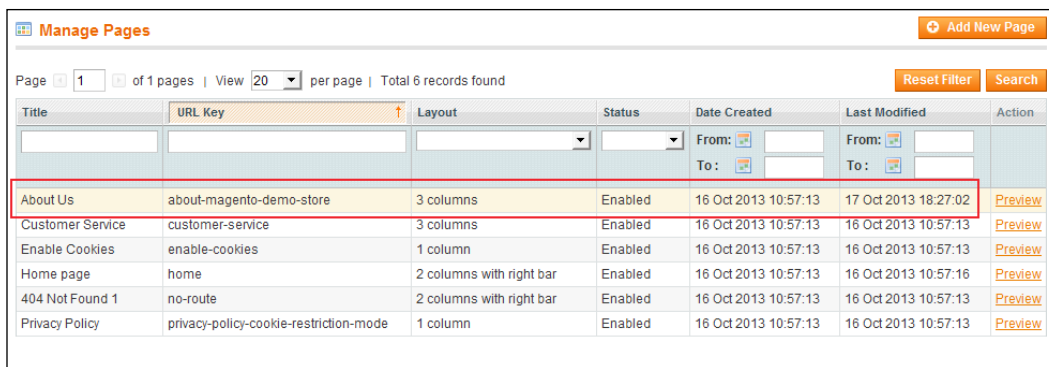
To put this CSS remedy in to action, open your Magento theme's `styles.css` file (it should be in the `/skin/frontend/default/responsive/css/` directory), and add the following CSS to it:

```
img,
img[height],
img[width]
{
 height: auto;
 max-width: 100%;
}
```

 This CSS doesn't need to be within one of your media queries, as you will (probably) want images in your store to resize to the width available regardless of where they appear!

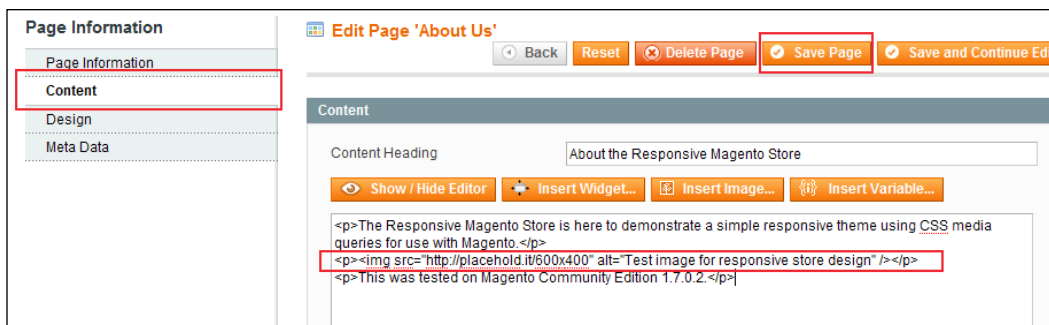
The `img[height]` and `img[width]` CSS selectors simply match any `img` elements in your page that have an assigned width or height attribute, and overwrite them. The `height: auto` CSS ensures that the height of the image remains in ratio to the width of the image. To put it simply, it prevents your images from becoming distorted when they're resized by the browser.

Next, log in to your Magento website's administration panel, and navigate to **CMS | Pages**. Select one of the pages to edit (the example uses the **About Us** page, as shown in the following screenshot):



| Title            | URL Key                                | Layout                   | Status  | Date Created         | Last Modified        | Action                  |
|------------------|----------------------------------------|--------------------------|---------|----------------------|----------------------|-------------------------|
| About Us         | about-magento-demo-store               | 3 columns                | Enabled | 16 Oct 2013 10:57:13 | 17 Oct 2013 18:27:02 | <a href="#">Preview</a> |
| Customer Service | customer-service                       | 3 columns                | Enabled | 16 Oct 2013 10:57:13 | 16 Oct 2013 10:57:13 | <a href="#">Preview</a> |
| Enable Cookies   | enable-cookies                         | 1 column                 | Enabled | 16 Oct 2013 10:57:13 | 16 Oct 2013 10:57:13 | <a href="#">Preview</a> |
| Home page        | home                                   | 2 columns with right bar | Enabled | 16 Oct 2013 10:57:13 | 16 Oct 2013 10:57:16 | <a href="#">Preview</a> |
| 404 Not Found 1  | no-route                               | 2 columns with right bar | Enabled | 16 Oct 2013 10:57:13 | 16 Oct 2013 10:57:13 | <a href="#">Preview</a> |
| Privacy Policy   | privacy-policy-cookie-restriction-mode | 1 column                 | Enabled | 16 Oct 2013 10:57:13 | 16 Oct 2013 10:57:13 | <a href="#">Preview</a> |

Once this page has loaded, select the **Content** tab in the left-hand column, and insert an image. You may find it useful to disable Magento's content editor toolbar using the **Show / Hide Editor** button towards the top of your screen, as shown in the following screenshot:



**Page Information**

- Page Information
- Content**
- Design
- Meta Data

**Edit Page 'About Us'**

Back Reset Delete Page **Save Page** Save and Continue Edit


**Content**

Content Heading: About the Responsive Magento Store

Show / Hide Editor Insert Widget... Insert Image... Insert Variable...

```
<p>The Responsive Magento Store is here to demonstrate a simple responsive theme using CSS media queries for use with Magento.</p>
<p></p>
<p>This was tested on Magento Community Edition 1.7.0.2.</p>
```

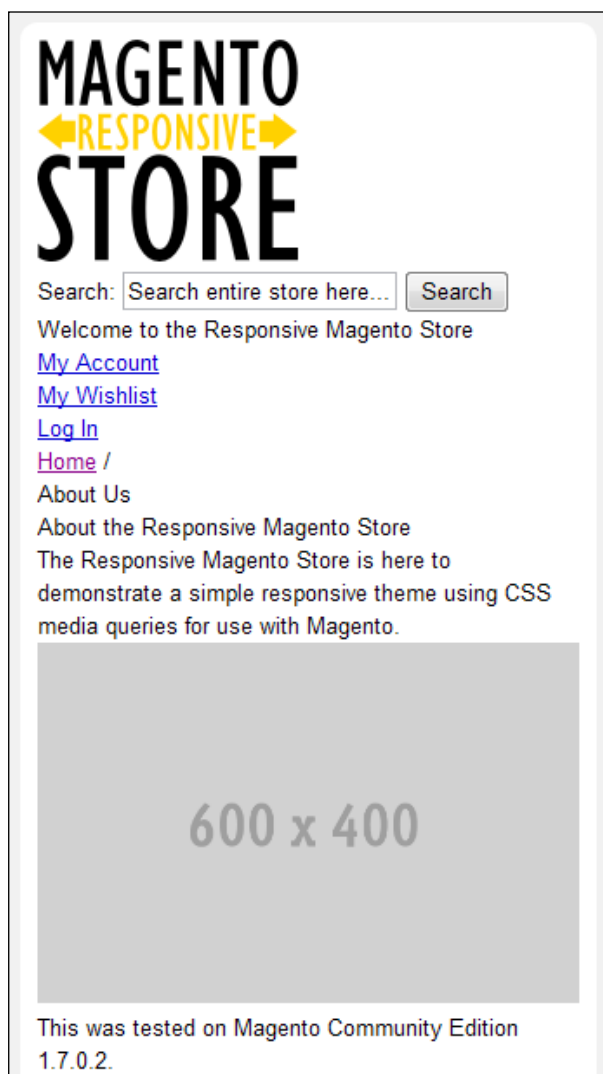


[  The example above uses `http://placeholder.it`, a free image placeholder service which will insert a grey image of the size you request in to your page. ]

Once you have added your image, click on the **Save Page** button in the top right of your screen and then view your page on the frontend of your website, as shown in the following screenshot:



If you view the Magento theme on your smartphone, or resize a suitable browser on your desktop (for example, Firefox, Chrome, or Opera), you should now see that the image is resized to fit the width available, as shown in the following screenshot:



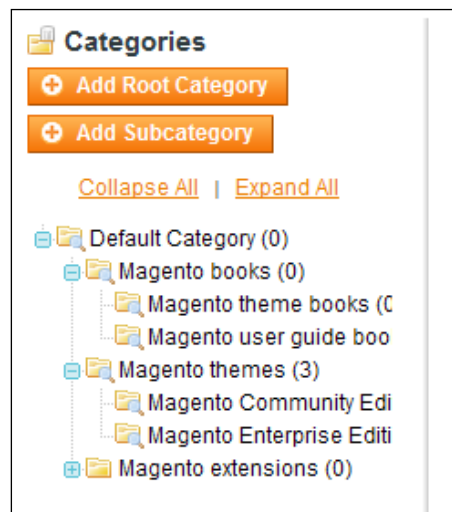
Images in your Magento store's content should now behave as you'd expect.

## Styling e-commerce navigation responsively in Magento

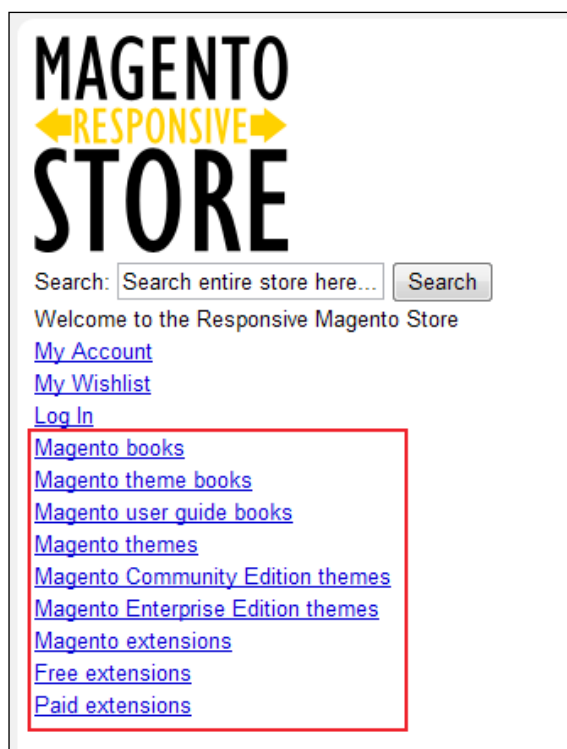
Navigation is one of the core features of any e-commerce website. If customers can't find products, they can't buy them! With the wide array of screen widths and browser capabilities to consider, this makes responsive navigation a challenge. There are two options this chapter considers:

- **Simple navigation:** It simply jumps the customer to the footer of the website that contains all of the key categories and pages they'll need
- **Advanced navigation:** It restyles the navigation for smaller devices to make items easier to select, and maintain Magento's dropdown styling for desktop users

Before you attempt any of these, ensure that your Magento store has some categories to navigate to in place. Do this by navigating to **Catalog | Manage Categories** in your Magento store's administration panel. You will need to ensure that you add the categories to the **Default Category**, and add a few categories each with their own subcategories, to replicate a similar structure to the one in the following screenshot that will be used in the example:



If you now view your Magento store's frontend, you should be able to see (unstyled) category navigation appear in the page, as shown in the following screenshot:



## Adding skip to footer navigation to your Magento store

This option is probably the easiest to implement for responsive navigation, and should provide the broadest range of support across different mobile and desktop browsers. To implement this option, you will need to perform the following steps:

1. Add a skip-to link in your store's header that is visible only on mobile/smaller-screened devices.
2. Add a static block in your store's footer to enable you to add the navigation you want your customers to see.

## Adding the skip-to link in your header template

Open your Magento theme's `header.phtml` file (in the `/app/design/frontend/default/responsive/template/page/html/` directory of your Magento installation) and do two things:

1. Hide the main Magento category navigation for mobile devices by wrapping it in a `div` element with a class of `mobile-hide`.
2. Add a link to skip to the footer.

The changes made to the `header.phtml` copied from Magento's fallback theme are highlighted in the following code:

```
<div class="header-container">
 <div class="header">
 <a href="<?php echo $this->getUrl('') ?>" title="<?php echo
 $this->getLogoAlt() ?>" class="logo">getLogoAlt()
 ?>" />
 <div class="quick-access">
 <?php echo $this->getChildHtml('topSearch') ?>
 <p class="welcome-msg"><?php echo $this->getWelcome() ?>
 <?php echo $this->getAdditionalHtml() ?></p>
 <?php echo $this->getChildHtml('topLinks') ?>
 <?php echo $this->getChildHtml('store_language') ?>
 </div>
 <?php echo $this->getChildHtml('topContainer'); ?>
 </div>
</div>

<div class="mobile-hide">
 <?php echo $this->getChildHtml('topMenu') ?>
</div>

<a class="mobile-nav" href="#footer-nav" title="Skip to store
 navigation"><?php $this->__('Skip to navigation'); ?>
```


In your theme's `styles.css` CSS file, add the following CSS above the desktop media query that you created:

```
.mobile-hide
{
 display: none;
}
.mobile-nav
{
```

```

background: #EFEFEF url("../images/mobile-nav.png") no-repeat
 center left;
border: 1px #CCC solid;
border-radius: 5px;
color: #333;
display: block;
padding: 5px 5px 5px 45px;
}
.mobile-nav:hover
{
 text-decoration: none;
}
.mobile-footer-nav
{
 background: #EFEFEF;
 clear: both;
 padding: 20px;
}
.mobile-footer-nav li
{
 display: inline;
 padding: 5px 10px;
}
.mobile-footer-nav li a
{
 color: #333;
}

```

 Ensure that the image `mobile-nav.png` in the code sample pack provided with this book is located in your theme's `/skin/frontend/default/responsive/images/` directory.

This CSS will hide any elements with this class applied from devices with smaller screens, and provide some basic styling to the mobile navigation button in the header, and the navigation links in the footer area too. Within your desktop media query, you will need to add the following CSS to ensure the elements hidden for mobile devices are shown for desktop visitors:

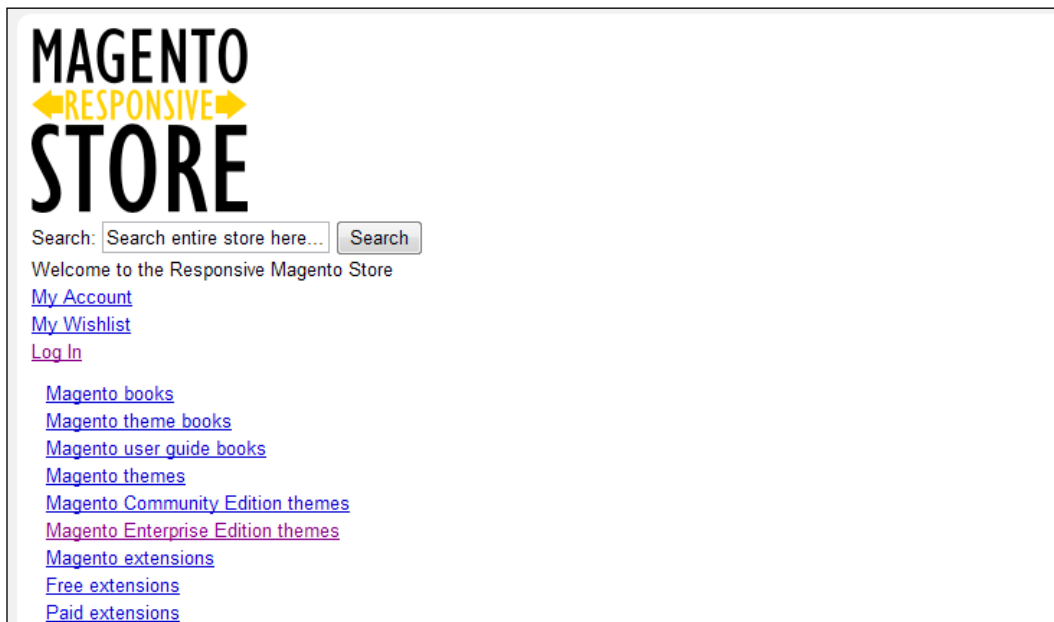
```

@media only screen and (min-width: 50em)
{
 .mobile-hide
 {
 display: block;
 }
}

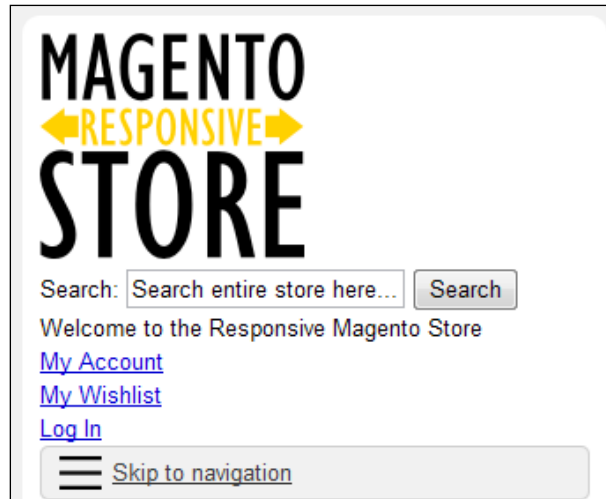
```

```
.mobile-nav,
.mobile-footer-nav
{
 display: none;
}
/* Other CSS in the media query */
}
```

If you view your Magento store's frontend at a desktop width, then the mobile navigation button isn't there; but you can see the main category navigation generated by Magento, as shown in the following screenshot:



If you view the new changes of your theme on a smartphone or smaller desktop width, you should see the new menu button appear, as shown in the following screenshot:

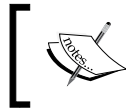


## Adding the footer sitemap navigation

The skip-to button now appears, but you will need to edit your theme's footer .phtml file to add a static block containing your store's navigation to finish this navigation. Copy the footer.phtml file from the app/design/frontend/base/default/template/page/html/ directory to the app/design/frontend/default/responsive/template/page/html/ directory, and then open it for editing, adding the content highlighted in the following code:

```
<div class="footer-container">
 <div class="footer">
 <?php echo $this->getChildHtml() ?>
 <p class="bugs"><?php echo $this->__('Help Us to Keep Magento
 Healthy') ?> - <a href="http://www.magentocommerce.com/
 bug-tracking" onclick="this.target='_blank'"><?php
 echo $this->__('Report All Bugs') ?> <?php echo
 $this->__('(ver. %s)', Mage::getVersion()) ?></p>
 <address><?php echo $this->getCopyright() ?></address>
 <div class="mobile-footer-nav" id="footer-nav">
 <?php echo $this->getLayout()->createBlock('cms/block')
 ->setBlockId('mobile-footer-nav')->toHtml() ?>
 </div>
 </div>
</div>
```





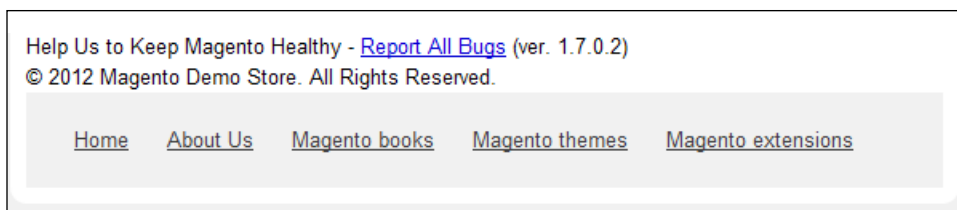
The ID attribute on the new `<div>` element you inserted will allow the **Skip to navigation** button, in the preceding screenshot, to know which element to skip down the page to when it is clicked.

You now need to create a static block in your Magento store. Log in to your administration panel, and navigate to **CMS | Static Blocks**. Click on the **Add New Block** button at the top right of your screen, and fill in the fields that appear to add your new static block. Make sure that the **Identifier** field matches the value in your reference to `footer.phtml` in the preceding code (the example uses `mobile-footer-nav` as the value).

The screenshot shows the 'Edit Block' interface in the Magento Admin. The title bar says 'Edit Block 'Mobile footer navigation''. At the top right are buttons: Back, Reset, Delete Block, Save Block, and Save and Continue Edit. The 'General Information' section contains the following fields:

- Block Title \*: Mobile footer navigation
- Identifier \*: mobile-footer-nav
- Status \*: Enabled
- Content \*: A 'Show / Hide Editor' button and a rich text editor. The editor contains a bulleted list:
  - [Home](#)
  - [About Us](#)
  - [Magento books](#)
  - [Magento themes](#)
  - [Magento extensions](#)

Click on the **Save Block** button towards the top right of your screen to save your new static block, and refresh your Magento store's frontend to see the new block appear for smaller-screened devices, as shown in the following screenshot:



That's it! Your simple mobile friendly navigation is ready and working!

## Dropdown navigation for your Magento store

An alternative method for navigation uses CSS within a media query to give customers with larger screens a more traditional dropdown navigation, while providing customers on smaller-screened devices with a more usable navigation for your product categories.



This option can work well for stores with small numbers of categories to list, but on larger Magento stores, you may find that this begins to crowd the actual content of your store!

Firstly, you will need to copy the default Magento CSS for dropdown navigation that starts with `.nav-container` or `#nav` (in `/skin/frontend/default/default/css/styles.css`) and adapt the style for your own responsive Magento theme within your desktop media query.

The adapted CSS for the example theme is provided in the following code:

```
@media only screen and (min-width: 50em)
{
 .nav-container
 {
 padding: 10px;
 }
 #nav
 {
 font-size:13px;
 margin:0 auto;
 padding:0 16px;
 width: 100%;
 }
 /* All Levels */
 #nav li
 {
 padding: 0;
 position:relative;
 text-align:left;
 }
 #nav li.over
 {
 z-index:998;
 }
 #nav a,
 #nav a:hover
```

```
{
 display:block;
 line-height:1.3em;
 text-decoration:none;
}
#nav span
{
 cursor:pointer;
 display:block;
 white-space:nowrap;
}
#nav li ul span
{
 white-space:normal;
}
/* 0 Level */
#nav li
{
 float:left;
}
#nav li.active a
{
 color:#d96708;
}
#nav a
{
 color:#333;
 float:left;
 padding:5px 12px 6px 8px;
 font-weight:bold;
}
#nav li.over a,
#nav a:hover
{
 color:#333;
}

/* 1st Level */
#nav ul li,
#nav ul li.active
{
 float:none;
 margin:0;
 padding-bottom:1px;
```

```
}
#nav ul li.last
{
 background:#EFEFEF;
 padding-bottom:0;
}

#nav ul a,
#nav ul a:hover
{
 background:none;
 float:none;
 padding:0;
}
#nav ul li a
{
 font-weight:normal !important;
}

/* 2nd Level */
#nav ul,
#nav div
{
 border:1px solid #CCC;
 position:absolute;
 width:15em;
 top:27px;
 left:-10000px;
}
#nav div ul
{
 border:none;
 position:static;
 width:auto;
}

/* 3rd+ Level */
#nav ul ul,
#nav ul div
{
 top:5px;
}

#nav ul li a
```

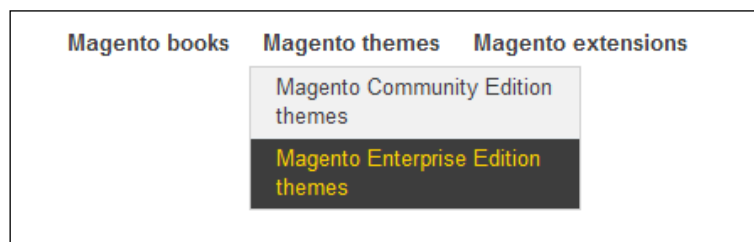
```
{
 background:#EFEFEF;
}
#nav ul li a:hover
{
 background:#333;
 color: #FC0;
}
#nav ul span,
#nav ul li.last li span
{
 padding:3px 15px 4px 15px;
}

/* Show menu */
#nav li ul.shown-sub,
#nav li div.shown-sub
{
 left:0;
 z-index:999;
}
#nav li .shown-sub ul.shown-sub,
#nav li .shown-sub li div.shown-sub
{
 left:100px;
}
/* Other CSS within the desktop media query */
} /* End of the desktop media query */
```



The preceding CSS is provided in the code pack that accompanies this book.

Once you have finished with this CSS, save your `styles.css` file to your Magento theme, and view your store's frontend at the desktop width you have defined, and you will see the dropdown style navigation appear, as shown in the following screenshot:



If you look at the frontend of your Magento store on a smaller screen width or on a smartphone device, you will not see any dropdown styling, as shown in the following screenshot:



You can style this simplistically by adding to your theme's `styles.css` file, above your media queries:

```
#nav a
{
 color: #333;
 text-decoration: none;
}
#nav a:hover
{
 text-decoration: underline;
}
#nav li,
#nav ul
{
 display: inline;
}
#nav li
{
```

```
padding: 10px;
}
#nav ul li a
{
color: #777;
}
```

Once saved, your navigation should be styled to be inline to preserve screen space, with subcategories appearing in a lighter grey colored text.

[Magento books](#) [Magento theme books](#) [Magento user guide books](#) [Magento themes](#) [Magento Community Edition themes](#) [Magento Enterprise Edition themes](#) [Magento extensions](#) [Free extensions](#) [Paid extensions](#)

Your simple responsive dropdown navigation is complete!

## Summary

Your new responsive Magento theme's basics are up and running now, with this chapter guiding you through:

- Creating the necessary directories for your Magento theme
- Enabling your new theme in Magento
- Adding media queries to your theme to establish breakpoints for different device widths
- Styling static pages in your Magento store
- Some of the available options for responsive navigation

The subsequent chapters will provide a more in-depth look at specific areas of your Magento store to theme.

# 2

## Making Your Store Responsive

So, you've now got the basics of your responsive Magento theme up and running, but there's still plenty to do! This chapter covers:

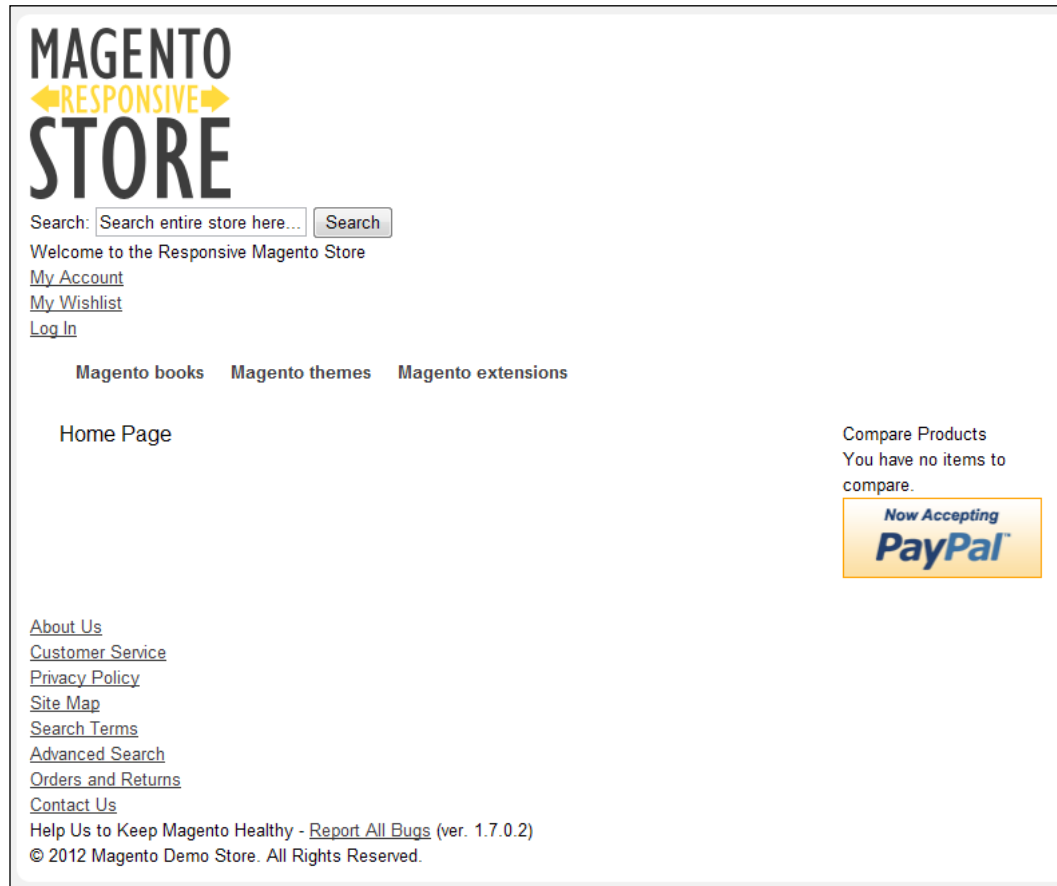
- Laying out your website's header and footer
- Responsive product page layout
- Dealing with product images for multiple devices
- Responsive category page layout
- Creating a responsive search results page

### **Laying out your website's header and footer**

The previous chapter covered some minimal layout, and now you can start altering the appearance of your Magento theme's header and footer to provide a more intuitive format for the content in them.



At the moment, your theme will look similar to the following screenshot:



Typically, the search feature is located to the top-left section of the screen, along with the **My Account** link and other related links.

## Adding the header and footer CSS

The most obvious place to begin styling your new Magento theme is the header area which will incorporate your store's logo, navigation, and logo. You can also begin to consider the footer area of your store which will appear on every page in your store's website.

Open your theme's `styles.css` file in the `skin/frontend/default/responsive/css` directory and add the following CSS outside any media queries (so that it will be used by all devices):

```
.breadcrumbs li {
 color: #777;
 display: inline;
}
.breadcrumbs a {
 color: #777;
}

.quick-access {
 color: #777;
 text-align: right;
}
.quick-access a {
 color: #777;
}
.nav-container {
 clear: both;
}
```

This provides styling to the breadcrumbs and search feature, which are key components of your ecommerce store in helping customers find what they're looking for.

## Adding CSS for larger screens

Next, you will need to add some further CSS to ensure it behaves as you'd expect on larger screens, so in the desktop media query in the `styles.css` file, add the following CSS code:

```
@media only screen and (min-width: 50em) {
 .logo {
 float: left;
 }
 /* Your other CSS */
}
```

If you now review the frontend of your Magento store, you will see things look neater, similar to the following screenshot:



## Adding CSS to change header and footer links

You can add the following CSS to your theme's `styles.css` file (outside any media queries again) to change the links in the top-right corner of the screen to display alongside each other:

```
.quick-access .links {
 list-style-type: none;
}
.quick-access .links li {
 display: inline;
}
```

You can treat the links in the footer of your Magento theme similarly, by adding the following relevant classes to the preceding CSS code:

```
.quick-access .links,
.footer ul {
 list-style-type: none;
}
.quick-access .links li,
.footer ul li {
 display: inline;
}
```

Finally, you can add the following CSS code:

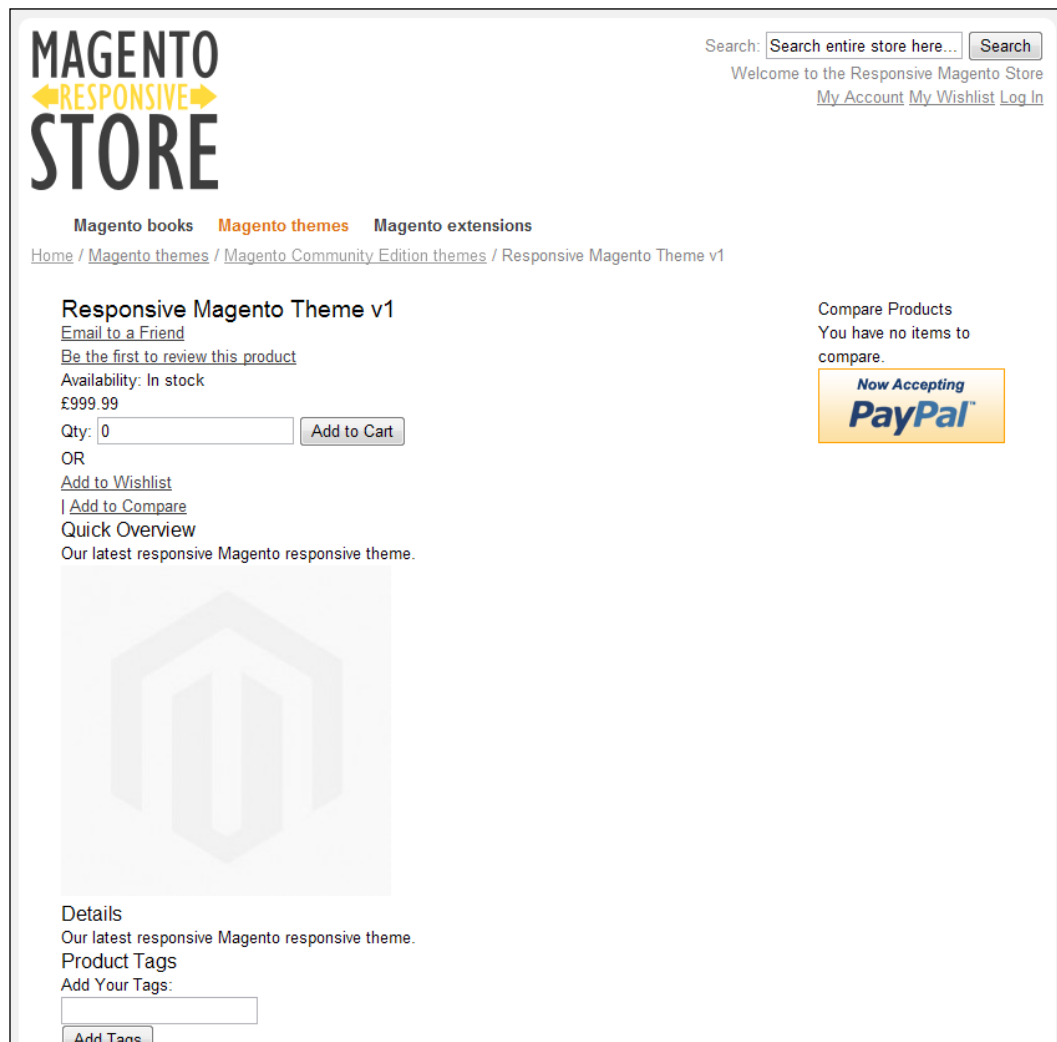
```
.footer {
 background: #333;
 border-radius: 10px;
 color: #fff;
 clear: both;
 padding: 10px;
 text-align: center;
}
.footer a {
 color: #fff;
}
```

If you view the frontend of your Magento website once again, you should see something similar to the following screenshot:



## Responsive product page layout in Magento

One of the key features of any ecommerce website is the product pages, and effort invested here can pay dividends on responsive websites. Ensure you have products added to your Magento store, and navigate to view one of your product pages. At the moment, it will look similar to the product page in the following screenshot on tablets and desktop computers:



## Laying out the product image and product information

The desired layout for a product page is typically with the primary product image to be displayed next to the product name, price, and description. To achieve this, you will need to copy the `app/design/frontend/base/default/template/catalog/product/view.phtml` file to the `app/design/frontend/default/responsive/template/catalog/product/view.phtml` file, and edit this file to add a class named `col-half` here:

```
<div class="product-shop col-half">
<div class="product-name">
 <h1><?php echo $_helper->productAttribute($_product,
 $_product->getName(), 'name') ?></h1>
</div>
```

In the same file, add the following class to the `<div>` tag that contains the product image:

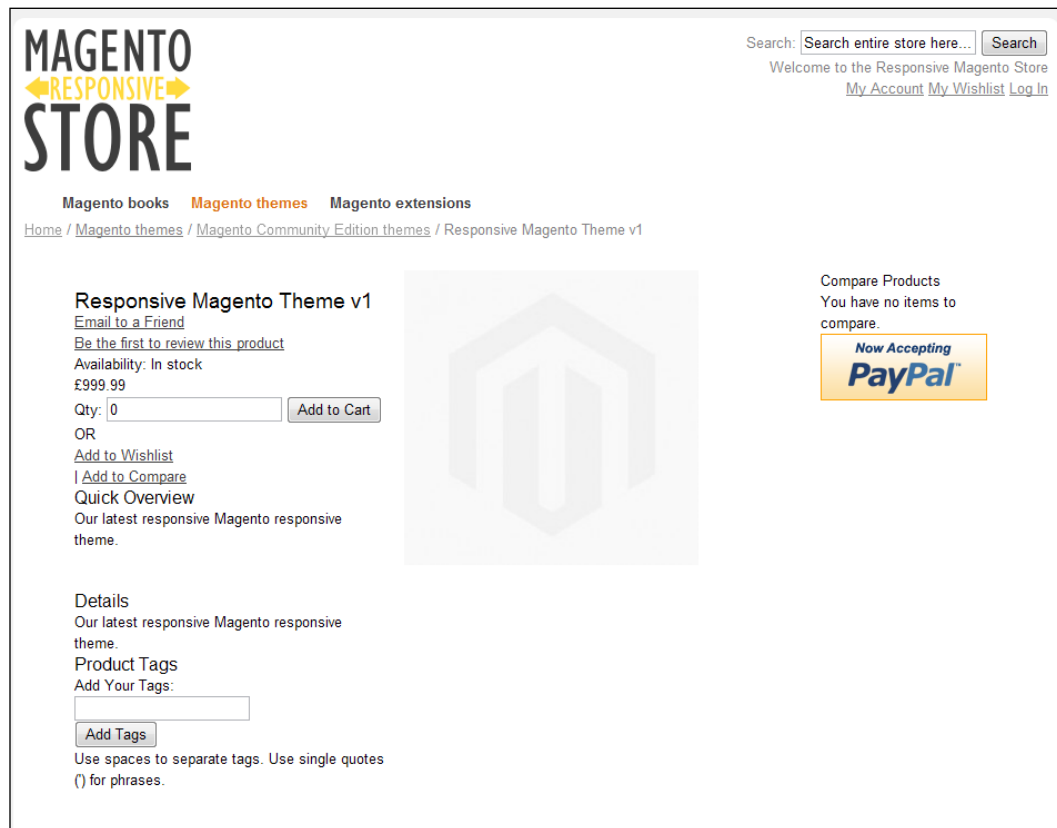
```
<div class="product-img-box col-half">
 <?php echo $this->getChildHtml('media') ?>
</div>
```

The final step here is to define some CSS for this class, so open your theme's `styles.css` file once more, and insert the following CSS highlighted in the following code within the media query:

```
.col-main,
.col-left,
.col-right,
.col-wrapper,
.col-half {
 display: inline;
 margin: 1%;
 padding: 2%;
}
.col-left,
.col-right,
.col-wrapper,
.col2-right-layout .col-main,
.col-half {
 float: left;
}
.col-half {
 width: 44%;
```

```
}
.col1-layout .col-main {
 float: none;
 width: 94%;
}
```

If you now preview your Magento store's frontend again, you should see the product page now looks more useful than before, as shown in the following screenshot:



## Making a definitive style

With a little more CSS, you can provide a more definitive style for the important elements of the product page outside of the media queries you have created in your `styles.css` file:

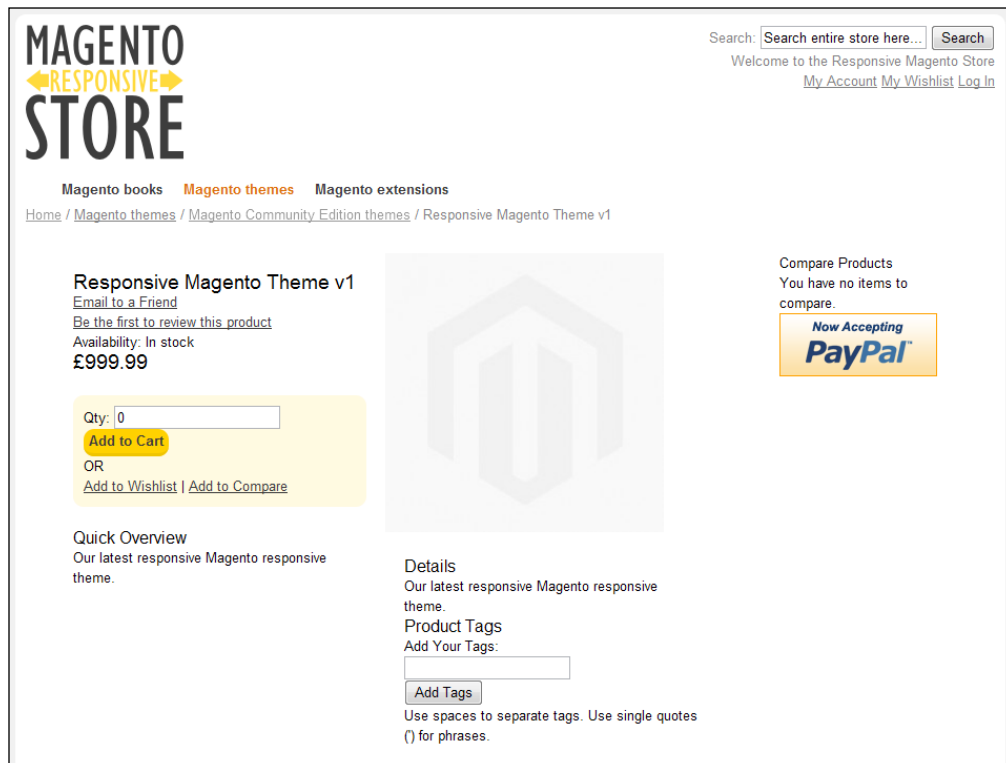
```
.add-to-box {
 background: #fff9de;
 border-radius: 10px;
```

```

margin: 20px 0;
padding: 10px;
}
.btn-cart {
background: #fc0;
border: none;
border-bottom: 3px #edbe00 solid;
border-radius: 10px;
color: #333;
font-weight: bold;
padding: 2px 5px;
}
.regular-price {
font-size: 150%;
}
.add-to-links li {
display: inline;
}

```

Your product page should now look similar to the following screenshot:





## Responsive category page layout

The category page, which lists related products, is a core component of many ecommerce websites. A well-designed category page allows your customers to find the product they're looking for more easily, and this is especially important when considering both mobile and desktop users viewing the same responsive page.

Magento provides two methods for your customers to view products:

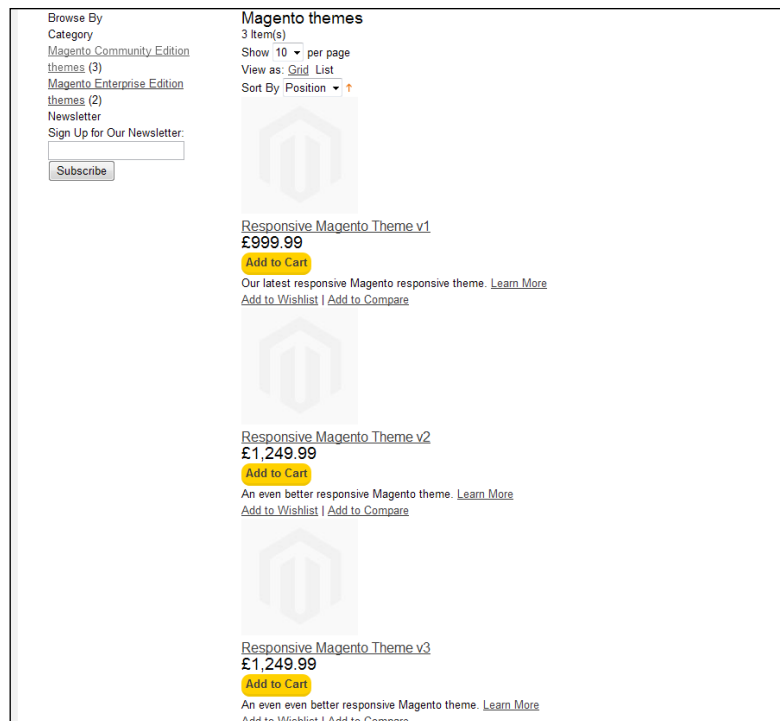
- **List view:** In this view, products are displayed one under the other
- **Grid view:** In this view, products are displayed in rows across the page, and then down the page



Before you begin with the category page, ensure that you have a number of products and categories added to your Magento store.

## Category list view

If you view your store's category page in the list view, you should currently see something similar to the following screenshot:



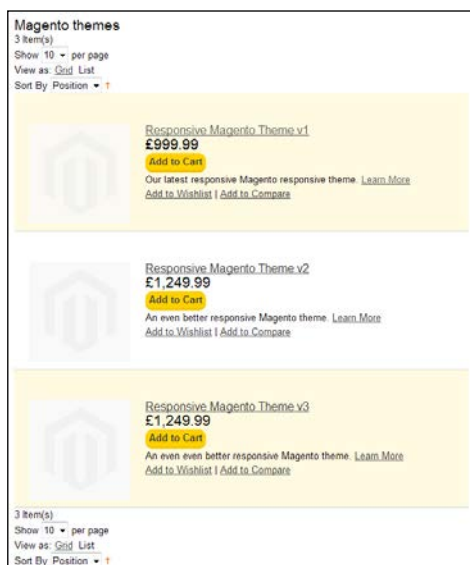
This is about what you should expect from the list view, but you can add CSS to alter the layout slightly. Open your theme's `styles.css` file (in the `skin/frontend/default/responsive/css` directory), and add the following CSS to alter the layout of the category products list outside of the desktop media query:

```
ol.products-list {
 list-style-type: none;
}
.products-list li.item {
 border-bottom: 1px #ccc solid;
 clear: both;
 display: block;
 padding: 40px 20px;
}
.products-list .product-image {
 display: block;
 float: left;
 margin-right: 20px;
}
```

Finally, you can target every other row of products in the list with the `odd` class:

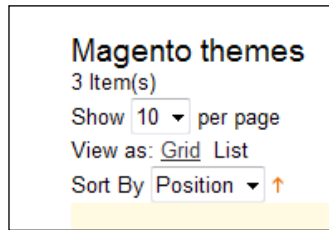
```
.products-list li.item.odd {
 background: #fff9de;
}
```


View the category list once you have saved your changes, and you should see the layout is now more sensible for browsing lists of products, as shown in the following screenshot:



## Product pager, pagination, and sort by dropdown

As you can see, the tools that control how many products are displayed, and how they are ordered, aren't yet styled, as shown in the following screenshot:



 You can control how many products are displayed in the Magento's administration panel by navigating to **System | Configuration | Catalog | Frontend**.

In the `styles.css` file, you can define the following generic style outside the desktop media query:

```
.toolbar {
 clear: both;
}
.pager, .sorter {
 background: #fff9de;
 border-radius: 10px;
 margin: 10px;
 padding: 10px;
}
.pager li {
 display: inline;
 padding: 10px;
}
```

Next, copy the `base/default/template/catalog/product/list/toolbar.phtml` file to the `default/responsive/template/catalog/product/list` directory, and open it for editing, adding the `col-half` class to the pager first:

```
<div class="pager col-half">
```

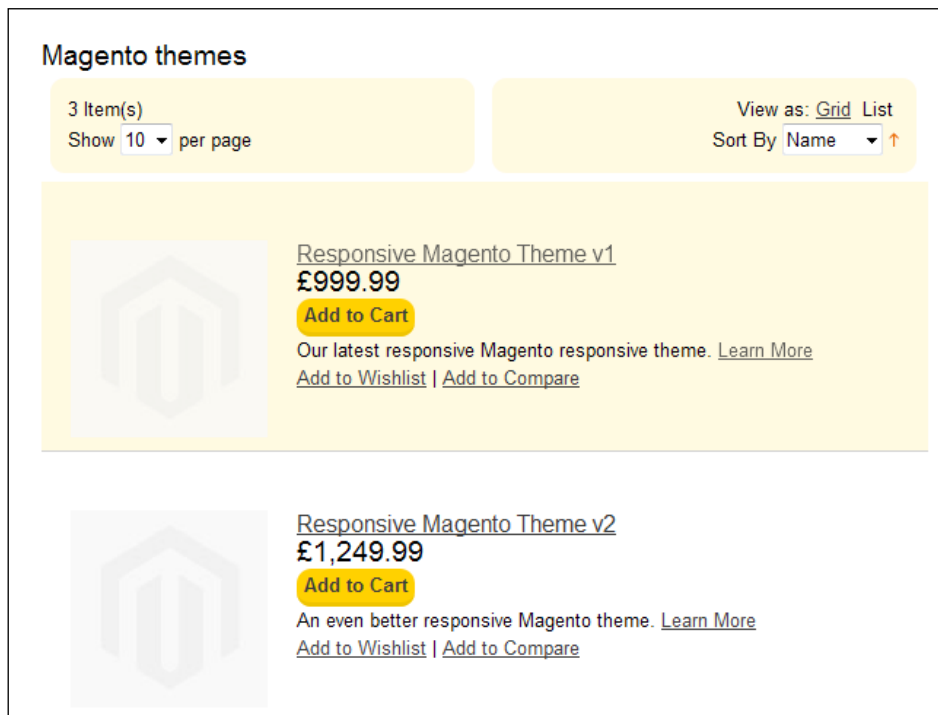
Next, add the `col-half` class to the sorter's `<div>` tag:

```
<div class="sorter col-half">
```

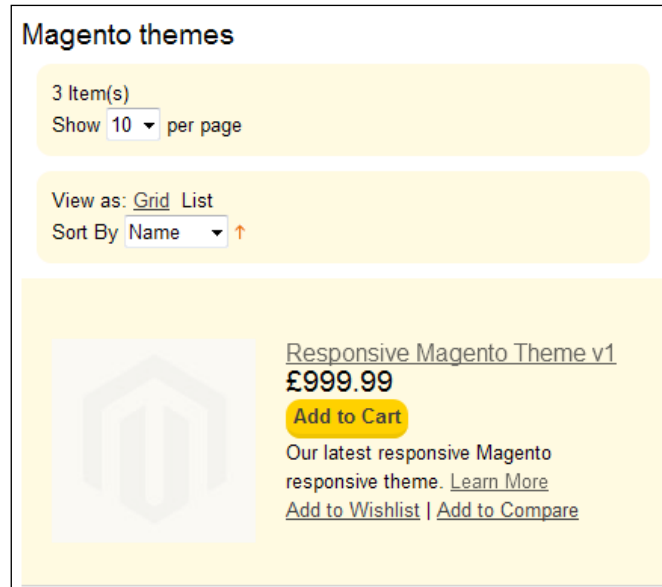
Within the media query for larger screens, you can add some additional layout information to help reposition these elements when more space is available:

```
.sorter {
 text-align: right;
}
```

View your category page now to see the effect these changes have made to your Magento store's design, as shown in the following screenshot:



On smaller screens, the sorter and pager blocks will be displayed above and below, rather than alongside each other, as shown in the following screenshot:

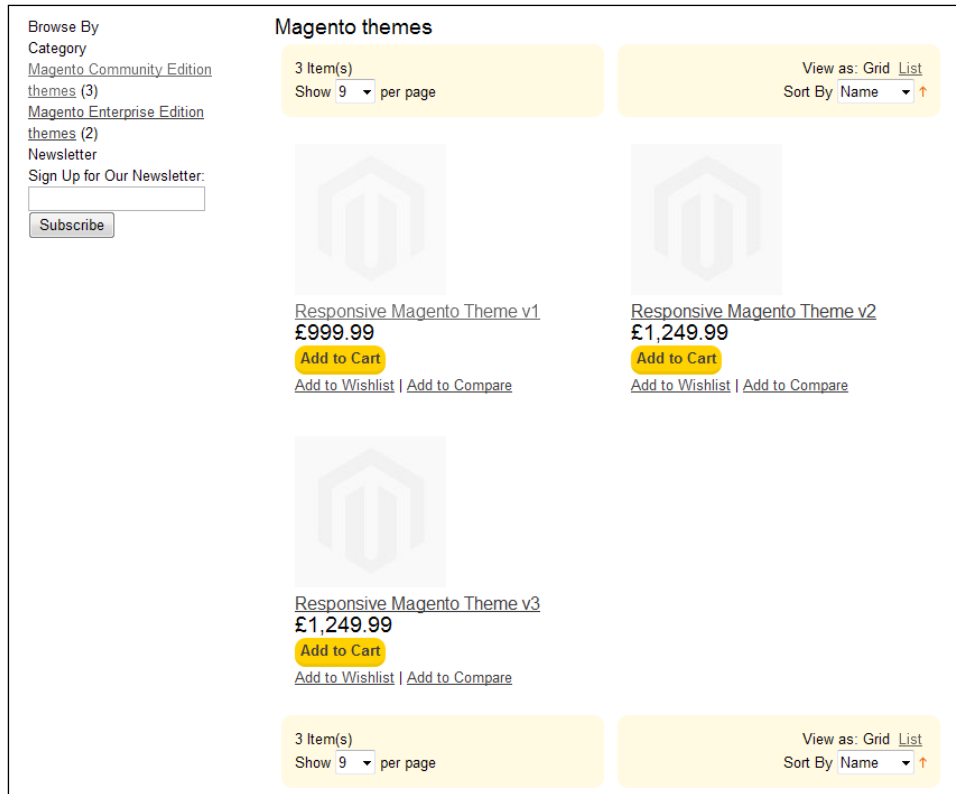


## Category grid view

Both Magento's list and grid view HTML are defined within the same file, so begin by copying the file at `app/design/frontend/base/default/template/catalog/product/list.phtml` to the `app/design/frontend/default/responsive/template/catalog/product` directory. Edit the file, and add the `col-half` class to each of the list items (`<li>`):

```
<?php if ($i++%$_columnCount==0): ?>
<ul class="products-grid">
<?php endif ?>
<li class="col-half item"><?php if (($i-1)%$_columnCount==0): ?>
 first<?php elseif ($i%$_columnCount==0): ?> last<?php endif; ?>
```

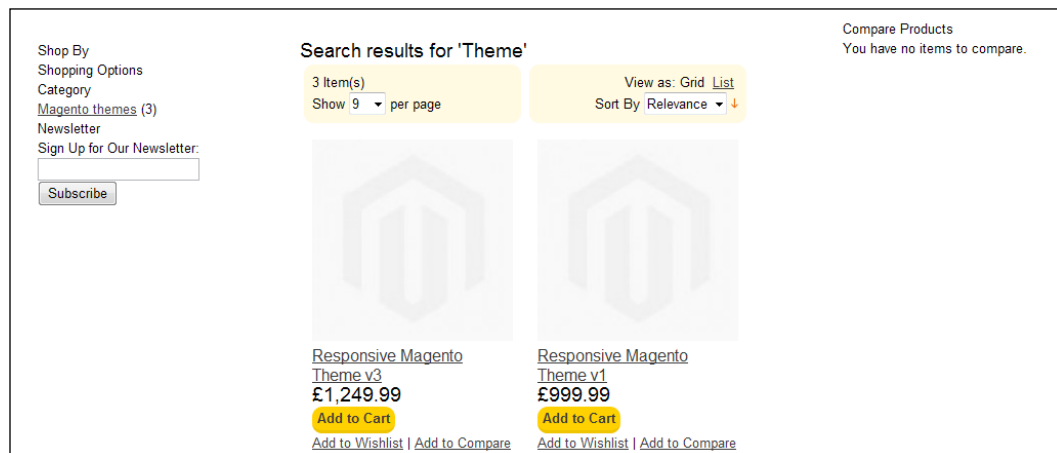
View the category page on the frontend of your Magento store now, ensuring you are seeing the products in the **Grid** mode, as shown in the following screenshot:



That's it: you now have a very simple responsive layout for your Magento store's category grid view!

## Dealing with Magento search results responsively

Search is another important component of your ecommerce website. By styling the category grid and category list for products in the preceding section, you have made a good start on styling the search results in Magento, which make use of the same template(s) by default. Try searching for some of your products to bring up some search results to demonstrate this, as shown in the following screenshot:



Can't see any results despite having added products to your Magento store? Try these tips: clear the search indexes by navigating to **System | Index Management**, and ensure your products are in stock and have a positive quantity assigned to them, if you have turned Magento's stock management feature on.

You can give more space to your Magento store's search results by assigning the one column layout to this page. Open your theme's `local.xml` file in the `/app/design/frontend/default/responsive/layout` directory.

## Resizing product images

This may be okay for your store, but you may want to ensure that the product images — an important element in these pages to encourage customers to view (and hopefully purchase) products from your store — are large enough to fill the width of screen available to them. To do this, revisit your theme's `list.phtml` file in the `/app/design/frontend/default/responsive/template/catalog/product/` directory and find an instance of the product image in it (which should look similar to the following code):

```
<a href="<?php echo $_product->getProductUrl() ?>" title="
 <?php echo $this->stripTags($this->getImageLabel
 ($_product, 'small_image'), null, true) ?>"
 class="product-image">
stripTags($this->
 getImageLabel($_product, 'small_image'), null, true) ?>" />

```



This appears twice: once in the code for the list view, and once for the grid view. Make sure you find both instances of the product image in this file!

The size of the image is defined three times in the preceding code snippet, and this is what you will need to change to alter the dimensions of product images in your category pages. The value you change this to will depend on the maximum size that the product image is likely to appear in your responsive Magento theme: the maximum size of the store is set in the following code snippet using the CSS's `max-width` property. As such, the following example changes the image's dimensions to a height and width of 200 pixels:

```
<a href="<?php echo $_product->getProductUrl() ?>"
 title="<?php echo $this->stripTags($this->getImageLabel
 ($_product, 'small_image'), null, true) ?>"
 class="product-image">
stripTags($this->getImageLabel
 ($_product, 'small_image'), null, true) ?>" />

```



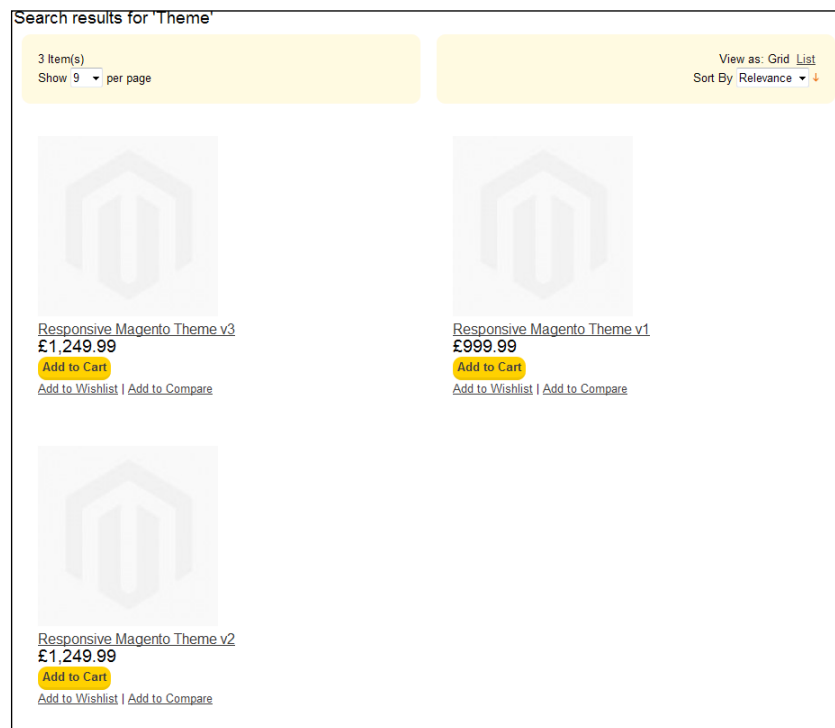
## Removing the height and width attributes

As in the preceding example, you can also remove the `height` and `width` attributes from the `<img>` element, as these can be a hindrance when resizing images in a responsive design.

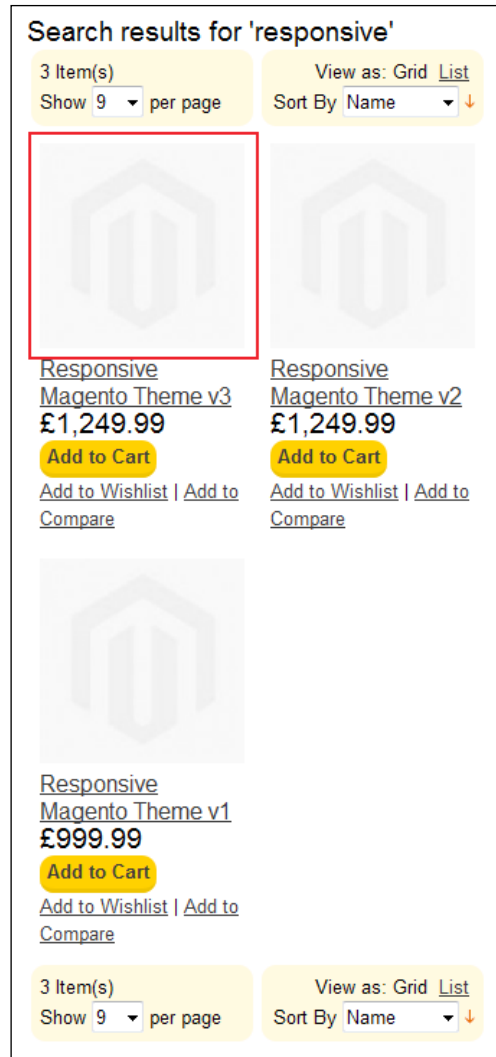
Next, you can set a maximum width on the page class in your theme's `styles.css` file in the `skin/frontend/default/responsive/css` directory, which will ensure that your store never becomes so wide that content is difficult to read or interpret (this already exists in your CSS file, so you will only need to add the two lines highlighted in the following code!):

```
.page {
 background: #fff;
 border-radius: 10px;
 margin: 10px auto; /* centre the store's design */
 padding: 10px;
 max-width: 1080px;
}
```

Once you have made these changes, view the frontend of your Magento store to see the updated styling for the search results:



As you resize your browser's window, the CSS in your theme's `styles.css` file which you defined earlier, applies `max-width: 100%` to the `<img>` element and ensures that the image is always resized to, at most, the width of the container (highlighted in red the following screenshot):



Some desktop browsers don't support media queries, though modern browsers such as Firefox, Chrome, Opera, and Safari do. See for <http://caniuse.com/css-mediaqueries> for native browser support of media queries.

## Summary

In this chapter, you have begun to develop your responsive Magento theme more deeply, concentrating on the Magento's category and product page templates. Specifically, by now you will have:

- Customized the general look and feel of your store's header and footer areas
- Added support for a responsive product page layout
- Dealt with product images in a basic manner to accommodate your responsive design
- Updated your Magento store's responsive category page layout template

The next chapter looks at customizing your Magento store's cart and checkout areas responsively.

# 3

## Responsive Checkout and Cart in Magento

Your Magento theme is coming together nicely now, but there's still plenty you can do to enhance other areas of your store, including the checkout and cart areas. This chapter covers:

- Styling Magento's shopping cart pages responsively
- Styling Magento's one page checkout responsively
- Styling customer account pages

### **Responsive shopping cart in Magento**

The shopping cart is the first port of call for your customers once they decide to purchase something from your store.

This makes your store's cart page an important step in the checkout process, and one that will require careful consideration to prevent it lowering your conversion rate. One of the biggest challenges for responsive cart pages on smaller devices is displaying the information required while keeping it clear and easy to both read and use.

At the moment, your theme's cart page will look similar to the following screenshot:

**MAGENTO**  
◀ RESPONSIVE ▶  
**STORE**


Search:

Welcome to the Responsive Magento Store  
[My Account](#) [My Wishlist](#) [Log In](#)

[Magento books](#) [Magento themes](#) [Magento extensions](#)

### Shopping Cart

Responsive Magento Theme v1 was added to your shopping cart.

Product Name	Unit Price	Qty	Subtotal
 Responsive Magento Theme v1	<a href="#">Edit</a> £999.99	<input type="text" value="1"/>	£999.99 <a href="#">Remove item</a>

#### Discount Codes

Enter your coupon code if you have one.

#### Estimate Shipping and Tax

Enter your destination to get a shipping estimate.

\*Country

State/Province

Zip/Postal Code

Subtotal £999.99  
Grand Total £999.99

[Checkout with Multiple Addresses](#)

## Styling form elements

As you can see, the cart page is not particularly attractive at the moment. You can start making it more attractive by styling form elements. Open your theme's `styles.css` file (in the `skin/frontend/default/responsive/css` directory) and add the following CSS to restyle the submit input types:

```
.button {
 background: #000;
```


```
border: none;
border-radius: 5px;
color: #fff;
padding: 5px 10px;
}
.button:active,
.button:focus {
background: #fc0;
color: #000;
}
.button:hover {
cursor: pointer;
}
.button:hover,
.button[disabled] {
opacity: 0.7;
}
.btn-checkout {
background: #fc0;
color: #000;
float: right;
font-weight: bold;
}
```

Next, you can style the text inputs and dropdowns by targeting the classes Magento provides. Add the following CSS to your theme's `styles.css` file outside the media queries you created earlier:

```
.col-main input, .col-main textarea, .col-main select {
margin-bottom: 20px;
}
input.input-text, .input-box select {
border: 1px #ccc solid;
border-radius: 5px;
padding: 5px;
}
```

If you save the changes to your Magento theme, you'll now see the form elements look more in-keeping with your new theme, as shown in the following screenshot:

### Shopping Cart

Product Name	Unit Price	Qty	Subtotal	
 Responsive Magento Theme v1	<a href="#">Edit</a> £999.99	<input type="text" value="1"/>	£999.99	<a href="#">Remove item</a>

[Continue Shopping](#) [Update Shopping Cart](#) [Clear Shopping Cart](#)

#### Discount Codes

Enter your coupon code if you have one.

  
[Apply Coupon](#)

#### Estimate Shipping and Tax

Enter your destination to get a shipping estimate.

\*Country

State/Province

Zip/Postal Code

  
[Get a Quote](#)

Subtotal £999.99  
Grand Total £999.99  
[Checkout with Multiple Addresses](#)

[Proceed to Checkout](#)

You can also add some separation between the various elements in the cart page to help provide a visual hierarchy in the page and give customers cues as to where they are in the checkout process. In your theme's `styles.css` file, add the following code:


```
.page-title,
#discount-coupon-form,
.shipping,
.data-table {
 margin-bottom: 20px;
}
```

```
.discount,

.shipping {
 background-color: #FFF9DE;
 border-radius: 10px;
 padding: 10px;
}
```

Your cart page will now look similar to the following screenshot:

## Shopping Cart

Product Name	Unit Price	Qty	Subtotal	
 <span>Responsive Magento Theme v1</span>	£999.99	1	£999.99	<a href="#">Edit</a> <a href="#">Remove item</a>

[Proceed to Checkout](#)

[Continue Shopping](#)
[Update Shopping Cart](#)
[Clear Shopping Cart](#)

### Discount Codes

Enter your coupon code if you have one.

[Apply Coupon](#)

### Estimate Shipping and Tax

Enter your destination to get a shipping estimate.

\*Country

United States

State/Province

Please select region, state or province

Zip/Postal Code

[Get a Quote](#)

Subtotal £999.99

Grand Total£999.99

[Checkout with Multiple Addresses](#)

[Proceed to Checkout](#)

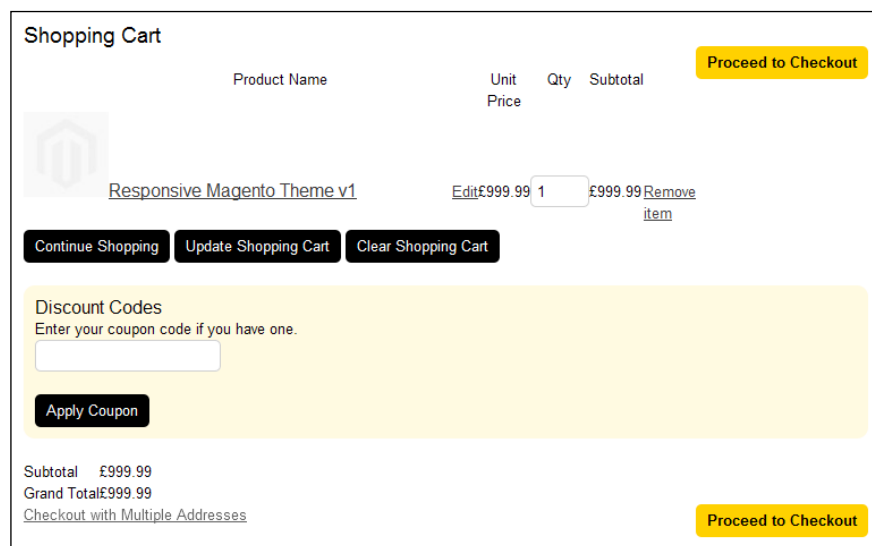


## Removing the shipping estimate tool from the Magento cart page

By default, Magento adds the shipping estimate tool at the cart stage: this is great if you want it, but not so useful if you offer free shipping on all orders, or a flat shipping fee. To remove this block, you will need to edit your theme's `local.xml` file in the `app/design/frontend/default/responsive/layout` directory and add the XML highlighted in the following code:

```
<?xml version="1.0"?>
<layout version="0.1.0">
 <!-- other Magento layout instructions -->
 <checkout_cart_index>
 <remove name="checkout.cart.shipping" />
 </checkout_cart_index>
</layout>
```

Once saved, if you view your Magento cart page again, you will see that the block has disappeared entirely, as shown in the following screenshot:



## Styling cart tables

Our penultimate step for styling the Magento cart page responsively is styling the table that contains the items your customer is buying. Once again in your theme's `styles.css` file, add the following CSS:

```
.data-table,
```

```


.cart form {
 clear: both;
 width: 100%;
}
.data-table th,
.data-table td {
 padding: 5px;
 vertical-align: top;
}
.data-table thead th {
 border-bottom: 1px #ccc solid;
}
.data-table th {
 font-weight: bold;
 text-align: left;
}
.data-table tbody tr:nth-of-type(even) td {
 background: #fff9De;
}

```

This styles the table to give it spacing inside each cell, making content easier to read, and shades every alternate product listed with a pale yellow color to help your customers differentiate long lists of products, as shown in the following screenshot:

Shopping Cart

Proceed to Checkout

Product Name	Unit Price	Qty	Subtotal
 <a href="#">Responsive Magento Theme v1</a>	<a href="#">Edit</a> £999.99	<input type="text" value="1"/>	£999.99 <a href="#">Remove item</a>

Continue Shopping

Update Shopping Cart

Clear Shopping Cart

Discount Codes

Enter your coupon code if you have one.

Apply Coupon

Subtotal £999.99

Grand Total£999.99

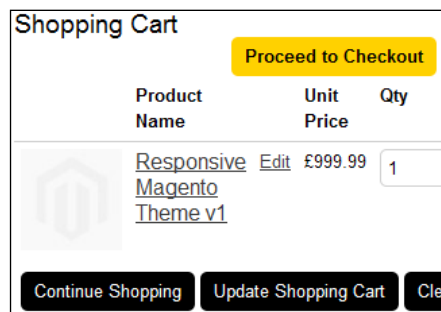
[Checkout with Multiple Addresses](#)

Proceed to Checkout

The CSS you just added will also shade every alternate product listed with a pale yellow color to help your customers differentiate long items of products.

## Removing unnecessary table columns for mobile and smaller screened devices

If you view the cart page on a device with a smaller viewport width, you will see that the cart's table content does not fit within the width of the screen available, as shown in the following screenshot:



One workaround you can apply to this situation is to hide unnecessary columns, and merge columns together. To do this, start by copying the `cart.phtml` file in the `app/design/frontend/base/default/template/checkout` directory to `app/design/frontend/default/responsive/template/checkout`.

Your first task is to merge the image and product name cells. To do this, find the following line in the `cart.phtml` file:

```
<table id="shopping-cart-table" class="data-table cart-table">
<col width="1" />
<col />
```

Now remove the second line, so that it reads the same as the following line:

```
<table id="shopping-cart-table" class="data-table cart-table">
<col />
```

Next, locate the following section in the file:

```
<?php $mergedCells = ($this->helper('tax')->
 displayCartBothPrices() ? 2 : 1); ?>
<thead>
<tr>
<th rowspan="<?php echo $mergedCells; ?>"> </th>
```

```
<th rowspan="<?php echo $mergedCells; ?>">
 <?php echo $this->__('Product Name') ?></th>
```

Remove the highlighted line so that it becomes the following code snippet:

```
<?php $mergedCells = ($this->helper('tax')->
 displayCartBothPrices() ? 2 : 1); ?>
<thead>
<tr>
<th rowspan="<?php echo $mergedCells; ?>">
 <?php echo $this->__('Product Name') ?></th>
```

Finally, copy the `app/design/frontend/base/default/template/checkout/cart/item/default.phtml` file to `app/design/frontend/default/responsive/template/checkout/cart/item/default.phtml`, and find the following lines which output the product image into the cart table:

```
<td>
<?php if ($this->hasProductUrl()):?><a href="
 <?php echo $this->getProductUrl() ?>" title="
 <?php echo $this->htmlEscape($this->getProductName()) ?>"
 class="product-image"><?php endif;?>htmlEscape
 ($this->getProductName()) ?>" /><?php if
 ($this->hasProductUrl()):?><?php endif;?>
</td>
```


Copy the preceding highlighted code, and remove the surrounding table cell elements (`<td>` and `</td>` from the file). Paste it in to the following table cell, so that it now looks similar to the following code snippet:

```
<td>
<?php if ($this->hasProductUrl()):?><a href="
 <?php echo $this->getProductUrl() ?>" title="
 <?php echo $this->htmlEscape($this->getProductName()) ?>"
 class="product-image"><?php endif;?>htmlEscape
 ($this->getProductName()) ?>" /><?php if
 ($this->hasProductUrl()):?><?php endif;?>
<h2 class="product-name">
<?php if ($this->hasProductUrl()):?>
<a href="<?php echo $this->getProductUrl() ?>">
 <?php echo $this->htmlEscape($this->getProductName()) ?>
<?php else: ?>
<?php echo $this->htmlEscape($this->getProductName()) ?>
<?php endif; ?>
</h2>
```

Once saved to your Magento theme, you should now see the slimmed-down version of your cart, as shown in the following screenshot:

### Shopping Cart

Proceed to Checkout

Product Name	Unit Price	Qty	Subtotal
 <a href="#">Responsive Magento Theme v1</a>	<a href="#">Edit</a> £999.99	<input type="text" value="1"/>	£999.99 <a href="#">Remove item</a>

Continue Shopping

Update Shopping Cart

Clear Shopping Cart

#### Discount Codes


Enter your coupon code if you have one.

Apply Coupon

On larger screens, the product name and image may require some styling as you can see in the following screenshot:

### Shopping Cart

Proceed to Checkout

Product Name	Unit Price	Qty	Subtotal
 <a href="#">Responsive Magento Theme v1</a>	<a href="#">Edit</a> £999.99	<input type="text" value="1"/>	£999.99 <a href="#">Remove item</a>

Continue Shopping

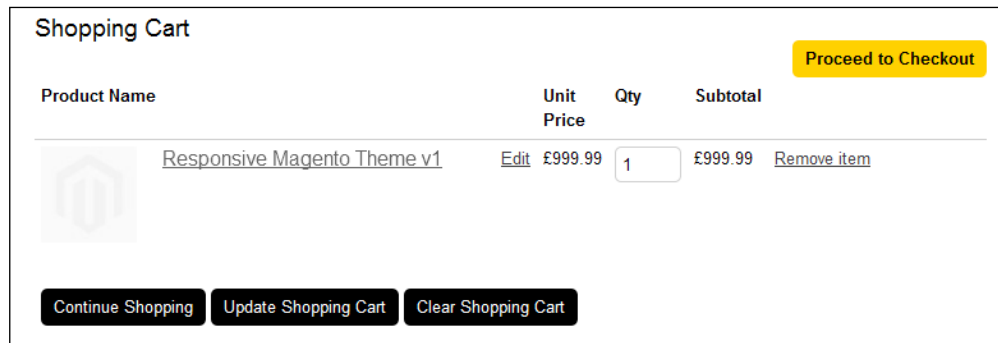
Update Shopping Cart

Clear Shopping Cart

Open your `styles.css` file and add the following code within your media query for tablet and desktop viewport widths:

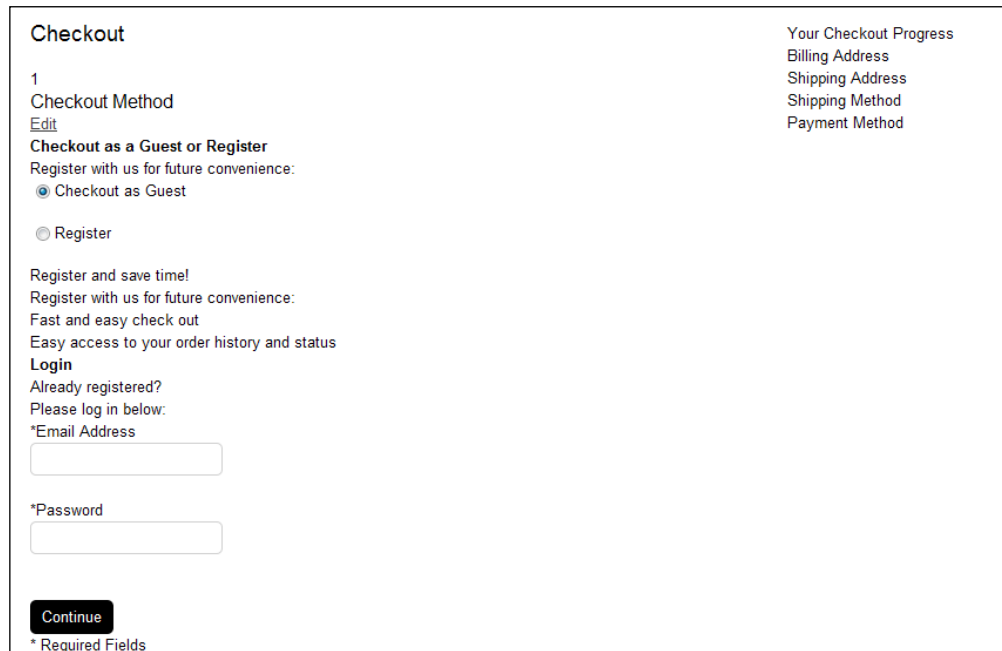
```
@media only screen and (min-width: 50em) {
 /* other CSS */
 .cart-table .product-image,
 .cart-table h2.product-name {
 display: inline;
 float: left;
 margin: 0 20px 20px 0;
 }
 .cart-table h2.product-name a {
 display: block;
 }
 .cart-table tbody td:first-of-type {
 min-width: 350px;
 }
}
```

That's it! The basics of your responsive Magento cart page are ready, as shown in the following screenshot:



# Responsive one page checkout in Magento

Your next step to create your responsive Magento theme is customizing the one page **Checkout**, which is the Magento's one page checkout. At the moment, you will see this is quite unstyled, as shown in the following screenshot:



The screenshot shows the Magento one-page checkout interface. On the left, under the heading "Checkout", there is a progress indicator "1" and the section "Checkout Method" with an "Edit" link. Below this, it says "Checkout as a Guest or Register" and "Register with us for future convenience:". There are two radio buttons: "Checkout as Guest" (selected) and "Register". Further down, it encourages registration with benefits like "Fast and easy check out" and "Easy access to your order history and status". A "Login" section follows, asking "Already registered?" and "Please log in below:" with input fields for "\*Email Address" and "\*Password". A "Continue" button is at the bottom of this section. On the right, a vertical list titled "Your Checkout Progress" shows the steps: "Billing Address", "Shipping Address", "Shipping Method", and "Payment Method". At the very bottom, a note says "\* Required Fields".

Open your theme's `styles.css` file in the `skin/frontend/default/responsive/css` directory and begin to style the one page checkout sections first. You can do by adding the following CSS outside any media queries you have previously created so it applies to all devices:

```
.opc .step {
 background: #fff9De;
 border-bottom-right-radius: 10px;
 border-bottom-left-radius: 10px;
 padding: 10px;
}
.opc .section {
 clear: both;
 margin-bottom: 20px;
}
```

```

.opc .step-title {
 background: #777;
 border-top-right-radius: 10px;
 border-top-left-radius: 10px;
 color: #fff;
 padding: 10px;
}
.opc .allow .step-title {
 background: #333;
}
.opc .active .step-title {
 background: #FC0;
 color: #333;
}
.opc .active .step-title a,
.opc .active .step-title h2 {
 color: #333;
}
.step-title h2, .step-title a {
 color: #fff;
 vertical-align: middle;
}
.step-title .number {
 background: #fff;
 border-radius: 50%;
 color: #333;
 float: left;
 margin: 0 10px 0 0;
 padding: 0 5px;
}
.step-title a {
 float: right;
}

```

This provides some layout and styling to the elements in the one page checkout to give the sections visual hierarchy and to make it clear which step the customer is currently viewing, and which of the previous steps they are able to revisit and change.



Columns not being displayed correctly? If you're only viewing, you will need to clear your floated items. Look in the `styles.css` file in the `skin/frontend/default/default/css` directory and you will see a section that begins with the comment: `/* Clears`. Copy this block of CSS to the bottom of your own theme's `styles.css` file.



Your next step defines some style for form elements within the one page checkout:

```
.wide .input-text {
 width: 95%;
}
input:focus, input:active, select:focus, select:active,
textarea:focus, textarea:active {
 border: 1px #fc0 solid;
}
.required em {
 color: #C00;
 margin-right: 5px;
}
```

If you save your progress and view your Magento store's one page checkout now, you will see it now looks more friendly to customers, as shown in the following screenshot:

The screenshot displays the Magento one-page checkout interface. At the top, the word "Checkout" is centered. On the right side, a vertical list titled "Your Checkout Progress" shows the steps: Billing Address, Shipping Address, Shipping Method, and Payment Method. The main content area is divided into three horizontal sections, each with a numbered step indicator and an "Edit" link.

- Step 1: Checkout Method** (Yellow background)
  - Checkout as a Guest or Register**: Includes a message "Register with us for future convenience:" and two radio buttons: "Checkout as Guest" (selected) and "Register". Below this, it says "Register and save time! Register with us for future convenience: Fast and easy check out Easy access to your order history and status". A "Continue" button is at the bottom left.
  - Login**: Includes a message "Already registered? Please log in below:" and two input fields: "Email Address" and "Password". A "Login" button is at the bottom right. A note "\* Required Fields" is above the "Forgot your password?" link.
- Step 2: Billing Information** (Gray background)
- Step 3: Shipping Information** (Gray background)

You can now style the **Your Checkout Progress** block which appears in the column on the right-hand side of your store. Once again in your theme's `styles.css` file, add the following CSS to begin styling the generic blocks that appear in Magento's sidebars:

```
.block {
 background: #fff9De;
 border-radius: 10px;
 margin-bottom: 20px;
 padding: 10px;
}
.block .block-title {
 border-bottom: 1px #fc0 solid;
 font-weight: bold;
 margin-bottom: 10px;
}
```

Once you have done this, you can begin to style the one page checkout progress block specifically by adding the following CSS to your theme's `styles.css` file:

```
.block {
 background: #fff9De;
 border-radius: 10px;
 margin-bottom: 20px;
 padding: 10px;
}
.block .block-title {
 border-bottom: 1px #fc0 solid;
 font-weight: bold;
 margin-bottom: 10px;
}
.opc-block-progress dt {
 font-weight: bold;
}
.opc-block-progress dd {
 margin-bottom: 10px;
}
```

Refreshing your store's checkout page and progressing through the first few initial steps, you will see that the progress block in the sidebar is now more distinct, as shown in the following screenshot:

**Checkout**

- 1 Checkout Method [Edit](#)
- 2 Billing Information [Edit](#)
- 3 Shipping Information [Edit](#)
- 4 Shipping Method [Edit](#)

Flat Rate  
☒ Fixed £5.00

[Back](#) [Continue](#)

**Your Checkout Progress**

**Billing Address | [Change](#)**  
Richard Carter  
Peacock Carter Ltd  
Adamson House  
Newcastle upon Tyne, NE1 1SG  
United Kingdom  
T: 0191 340 7158

**Shipping Address | [Change](#)**  
Richard Carter  
Peacock Carter Ltd  
Adamson House  
Newcastle upon Tyne, NE1 1SG  
United Kingdom  
T: 0191 340 7158

**Shipping Method**  
**Payment Method**

On devices with smaller screens, the one page checkout defaults to a one-column layout, as shown in the following screenshot:

\* Country  
United Kingdom

\* Telephone

Fax

☒ Ship to this address  
☐ Ship to different address

\* Required Fields  
[Continue](#)

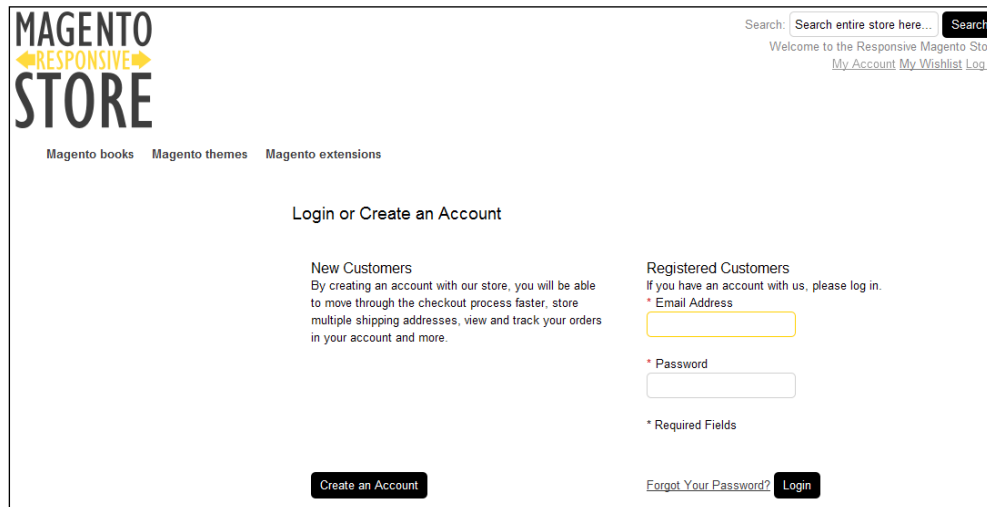
- 3 Shipping Information [Edit](#)
- 4 Shipping Method [Edit](#)
- 5 Payment Information [Edit](#)

## Styling Magento customer account pages responsively

Previously in this chapter, you applied a general style to form elements, so part of your work on the customer account pages is done already!

### Setting the account login page template to a one-column layout

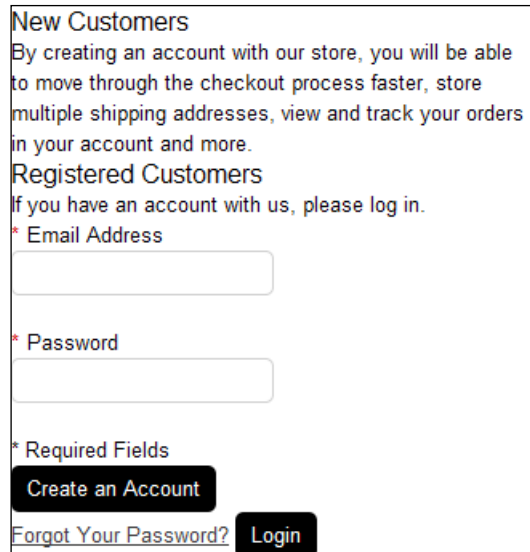
At present, your login page for customers will look similar to the following screenshot:



You can make better use of the space here by adding some CSS within your desktop/larger screen media query. This ensures that pages with the one-column layout assigned to them display the main column of content, the full width of the screen, and can be done by adding the following CSS to your theme's `styles.css` file:

```
@media only screen and (min-width: 50em) {
 /* Other CSS */
 .coll-layout .col-main {
 float: none;
 width: 94%;
 }
 .customer-account-login .page-title {
 text-align: center;
 }
}
```

The preceding code fills the screen for devices with larger screens, but the default login screen does not work well for small screen devices because the **Login** and **Register** buttons do not appear directly under their respective forms, as shown in the following screenshot:



You can fix this by copying the file at `app/design/frontend/base/default/template/persistent/customer/form/login.phtml` to `app/design/frontend/default/responsive/template/persistent/customer/form/login.phtml`. Once you have done this, open the file for editing, and move the buttons so that the registration block looks similar to the following code:

```
<div class="col-1 new-users">
<div class="content">
<h2><?php echo $this->__('New Customers') ?></h2>
<p><?php echo $this->__('By creating an account with our store,
 you will be able to move through the checkout process faster,
 store multiple shipping addresses, view and track your orders in
 your account and more.') ?></p>
</div>
<div class="buttons-set">
 <button type="button" title="<?php echo $this->__
 ('Create an Account') ?>" class="button" onclick="window.location='
 <?php echo Mage::helper('persistent')->getCreateAccountUrl
 ($this->getCreateAccountUrl()) ?>';">
 <?php echo $this->__('Create an Account') ?></button>
</div>
</div>
```

Similarly, your login section should look similar to the following code now:

```
<div class="col-2 registered-users">
<div class="content">
<h2><?php echo $this->__('Registered Customers') ?></h2>
<p><?php echo $this->__('If you have an account with us, please
 log in.') ?></p>
<ul class="form-list">

<label for="email" class="required">*
 <?php echo $this->__('Email Address') ?></label>
<div class="input-box">
<input type="text" name="login[username]" value="
 <?php echo $this->htmlEscape($this->getUsername()) ?>"
 id="email" class="input-text required-entry validate-email"
 title="<?php echo $this->__('Email Address') ?>" />
</div>

<label for="pass" class="required">*<?php echo
 $this->__('Password') ?></label>
<div class="input-box">
<input type="password" name="login[password]" class="input-text
 required-entry validate-password" id="pass" title="
 <?php echo $this->__('Password') ?>" />
</div>

<?php echo $this->getChildHtml('form.additional.info'); ?>
<?php echo $this->getChildHtml('persistent.remember.me'); ?>

<?php echo $this->getChildHtml('persistent.remember.me.tooltip');
 ?>
<p class="required"><?php echo $this->__('* Required Fields')
 ?></p>
<div class="buttons-set">
<a href="<?php echo $this->getForgotPasswordUrl() ?>"
 class="f-left"><?php echo $this->__('Forgot Your Password?') ?>
<button type="submit" class="button" title="<?php echo
 $this->__('Login') ?>" name="send" id="send2">
 <?php echo $this->__('Login') ?></button>
</div>
</div>
</div>
```

Once you have uploaded this file and refreshed the page, you'll see that the relevant button(s) have now appeared under the corresponding area, as shown in the following screenshot:

**Login or Create an Account**  
  
**New Customers**  
By creating an account with our store, you will be able to move through the checkout process faster, store multiple shipping addresses, view and track your orders in your account and more.  
**Create an Account**  
  
**Registered Customers**  
If you have an account with us, please log in.  
\* Email Address  
  
  
\* Password  
  
  
\* Required Fields  
[Forgot Your Password?](#) **Login**

## Styling error messages

If you attempt to login and provide an incorrect e-mail address or password, you will be presented with the following error message:

**Login or Create an Account**  
  
Invalid login or password.  
  

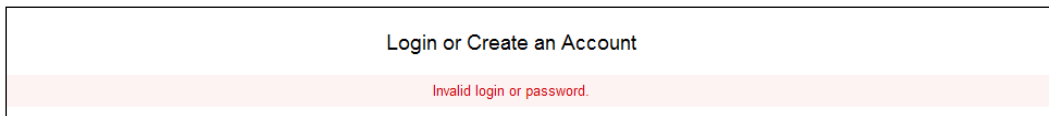
**New Customers**  
By creating an account with our store, you will be able to move through the checkout process faster, store multiple shipping addresses, view and track your orders in your account and more.  
**Create an Account**

**Registered Customers**  
If you have an account with us, please log in.  
\* Email Address  
  
  
\* Password  
  
  
\* Required Fields  
[Forgot Your Password?](#) **Login**

This is okay, but a little styling added to your theme's `styles.css` file can give it more prominence for customers, so they understand there's a problem. To do this, add the following code snippet to your theme's stylesheet:

```
.messages {
 clear: both;
}
.error-msg {
 background: #fdf2f2;
 border-radius: 10px;
 color: #C00;
 margin-bottom: 5px;
 padding: 5px;
 text-align: center;
}
.col-main .messages ul ul {
 margin: 0;
}
```

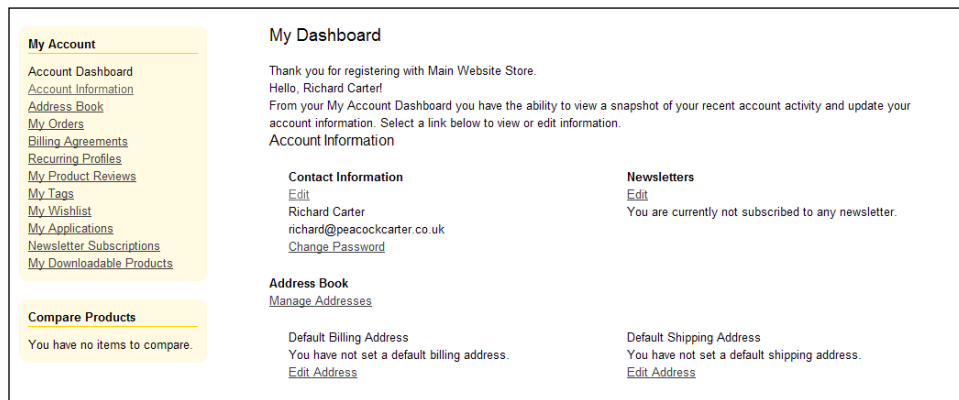
If you now fail logging in again, you'll see the message is more prominent on the page now, as shown in the following screenshot:





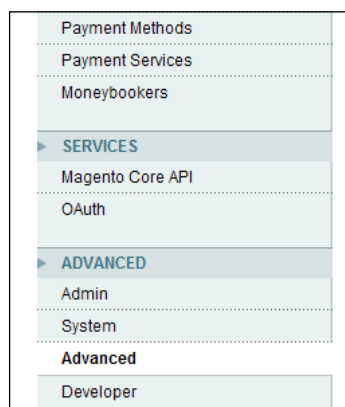
## Removing the My Applications and My Downloadable Products links from the Magento customer account area

For the next stage, you will need to have created a customer account on your Magento store, and ensure you are logged in to the customer account area, as shown in the following screenshot:

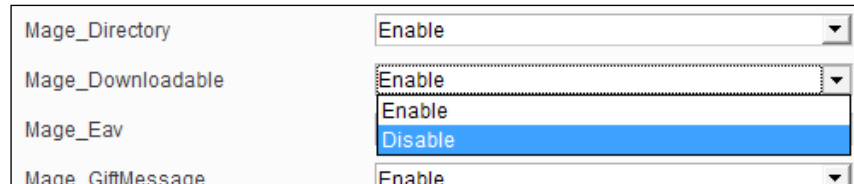


As you can see, the basic layout and styling are there already for the account pages, but there may be a few links in the **My Account** block in the left-hand side column that you may wish to remove, such as the **My Applications** and **My Downloadable Products** links.

You can remove some through Magento's backend, so navigate to **System | Configuration** in your Magento store's administration panel, and select the **Advanced** tab in the left-hand side column, as shown in the following screenshot:



In the **Disable Modules Output** section, you can set the drop-down value to **Disable** for the modules you wish to disable, as shown in the following screenshot:



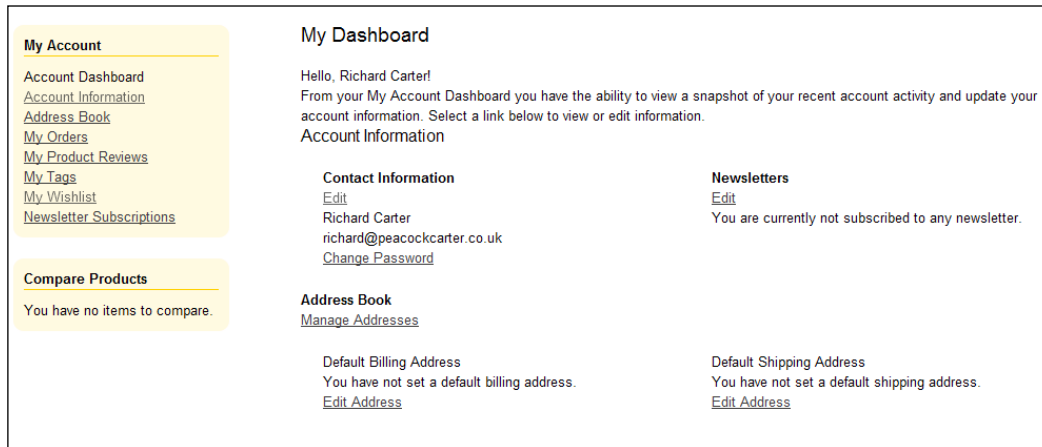
The **My Applications**, **Recurring Profiles**, and **Billing Agreements** links will remain (even if you disable the **Mage\_OAuth** module). To disable these links from your accounts menu, create the following blank files in your `app/design/frontend/default/responsive/layout/sales` directory:

- `billing_agreement.xml`
- `recurring_profile.xml`

This will remove the **Billing Agreements** and **Recurring Profiles** links respectively. To remove the **My Applications** link, you will need to copy the `oauth.xml` file from the `app/design/frontend/base/default/layout` directory to the `app/design/frontend/default/responsive/layout` directory. You will then need to find the following code in this file:

```
<customer_account>
<reference name="customer_account_navigation">
<action method="addLink" translate="label" module="oauth">
<name>OAuth Customer Tokens</name>
<path>oauth/customer_token</path>
<label>My Applications</label>
</action>
</reference>
</customer_account>
<oauth_customer_token_index translate="label">
<label>Customer My Account My OAuth Applications</label>
<update handle="customer_account"/>
<reference name="my.account.wrapper">
<block type="oauth/customer_token_list"
 name="oauth_customer_token_list"
 template="oauth/customer/token/list.phtml"/>
</reference>
</oauth_customer_token_index>
```

Now remove it entirely from your theme's `oauth.xml` file. Save this file, and the links will be removed once you refresh your Magento store's customer account pages, as shown in the following screenshot:



## Summary

This chapter developed your new responsive Magento theme further, adding customizations for:

- Styling Magento's shopping cart pages responsively
- Styling Magento's one page checkout responsively
- Styling customer account pages

The final chapter will help to guide you through some further enhancements that you can make to your Magento store to help customers on both small and large-screened devices.

# 4

## Enhancing Your Responsive Magento Theme

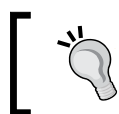
You've now laid the basics of your responsive Magento theme; you can now go about enhancing it for your customers. This chapter covers:

- Including `css3-mediaqueries.js` to better support older versions of Internet Explorer
- Improving the data entry at checkout for mobile customers
- Using the CSS `@font-face` rule to use custom fonts in your Magento theme

While not strictly necessary for your responsive Magento theme, these tasks can make your customers' experience of your store much easier and more enjoyable, which will hopefully lead to better yields from your website!

### Supporting CSS media queries in Internet Explorer with `css3-mediaqueries.js`

By default, earlier versions of Internet Explorer (8 and before) do not support the CSS media queries that you have used to make your Magento theme responsive.



For more information on browser support of CSS media queries, see <http://caniuse.com/css-mediaqueries>.

If you view your store in Internet Explorer 8 or earlier at the moment, you will see the following screenshot:



As the CSS relating to widths and layout of columns is within the media query for larger-screened devices, and the browser doesn't understand media queries, this CSS is ignored and no layout (or other styling within the media query) is applied to the page.

You can rectify this by including a JavaScript library called `css3-mediaqueries.js` to your Magento theme. Firstly, go to <https://code.google.com/p/css3-mediaqueries-js/downloads/list> and download the `css3-mediaqueries.js` file. Save this to the `skin/frontend/default/responsive/js` directory.

It's worth noting that the `css3-mediaqueries.js` library only supports the `@media` type and not media queries provided in the following format:

```
<link rel="stylesheet" type="text/css" href="style.css"
 media="screen and (min-width: 50em)">
```

## Adding a JavaScript file to your Magento theme through local.xml

Open your theme's `local.xml` file in the `app/design/frontend/default/responsive/layout` directory and locate the `<default>` handle which applies its changes to every page on your Magento store. Save the following code in your theme's `local.xml` file:

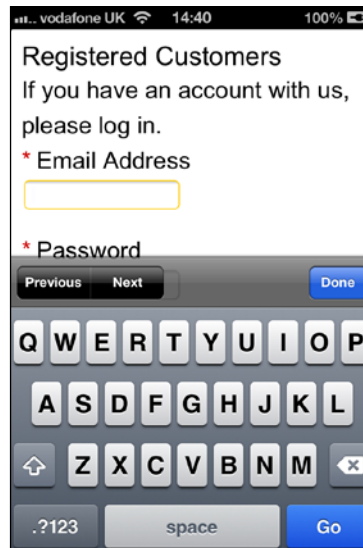
```
<?xml version="1.0"?>
<layout version="0.1.0">
 <default>
 <reference name="head">
 <action method="addItem">
 <type>skin_js</type>
 <name>js/css3-mediaqueries.js</name>
 <params/><if>lt IE 7</if>
 </action>
 </reference>
 </default>
 <!-- other layout -->
</layout>
```

If you now refresh your Magento theme in Internet Explorer 8, you'll see that styling within the media query appears, providing columns for the layout and the additional styling for the category navigation dropdowns, as shown in the following screenshot:



## Improving Magento store data entry for customers

One of the many potential barriers to mobile or smaller-screened devices being used to complete e-commerce orders is data entry. For example, entering your e-mail address when logging in on a mobile device such as an iPhone displays the following keypad:



This is okay. The user can find all of the keys they need, but it's not as efficient as it could be. HTML5 introduced new input types which, on more modern devices, provide a keyboard tailored to its use. So, for this field, you could change the input type to be `type="email"` rather than the (default) `type="text"`.

## Changing your Magento theme's doctype to HTML5

The first step in this process is to change your Magento theme's HTML doctype from the default. To do this, copy the skeleton template files from the `app/design/frontend/base/default/template/page/html` directory to the `app/design/frontend/default/responsive/template/page/html` directory. These should include:

- `1column.phtml`
- `2columns-left.phtml`

- 2columns-right.phtml
- 3columns.phtml

In each of these files, locate the following two lines that read:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="<?php echo
 $this->getLang() ?>" lang="<?php echo $this->getLang() ?>">
```

Change these lines to read the following code:

```
<!DOCTYPE html>
<!--[if lt IE 7]><html lang="<?php echo $this->getLang() ?>"
 class="no-js lt-ie9 lt-ie8 lt-ie7"><![endif]-->
<!--[if IE 7]><html lang="<?php echo $this->getLang() ?>"
 class="no-js lt-ie9 lt-ie8"><![endif]-->
<!--[if IE 8]><html lang="<?php echo $this->getLang() ?>"
 class="no-js lt-ie9"><![endif]-->
<!--[if gt IE 8]><!--><html lang="<?php echo $this->getLang() ?>"
 class="no-js"><!--<![endif]-->
```



Doing this also allows you to address independent versions of Internet Explorer using the class assigned to the html element.

This changes your Magento theme's doctype to HTML5, which now allows you to make use of the new input types available.

## Changing the input type value for login

To change the login screen's e-mail field input type, you will need to edit the login.phtml file in the app/design/frontend/default/responsive/template/persistent/customer/form directory. Locate the current e-mail address field for customers logging in, which should look similar to the following code snippet:

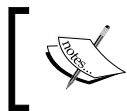
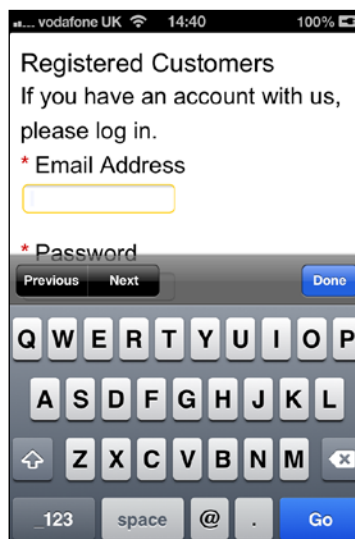
```
<label for="email" class="required">*
 <?php echo $this->__('Email Address') ?></label>
<div class="input-box">
<input type="text" name="login[username]" value="
 <?php echo $this->htmlEscape($this->getUsername()) ?>"
 id="email" class="input-text required-entry validate-email"
 title="<?php echo $this->__('Email Address') ?>" />
</div>
```



Change the `type` attribute on the input element here to read email:

```
<input type="email" name="login[username]" value="
 <?php echo $this->htmlEscape($this->getUsername()) ?>"
 id="email" class="input-text required-entry validate-email"
 title="<?php echo $this->__('Email Address') ?>" />
```

If you now save this file and attempt to edit the field's content on a smartphone or tablet device which supports HTML5 input types, you should see a slightly altered keypad better suited to entering an e-mail address, as shown in the following screenshot:



Devices that don't support the `input type="email"` attribute simply revert the type back to `input type="text"`, which is supported.

## Changing the input type value for registration

You're able to do this for the customer registration form on your Magento store too. You can make similar substitutes throughout the forms in your Magento theme. Some useful input types in addition to the `type="email"` attribute introduced in the previous section that you might use are:

- `type="search"`: Used for search query fields. This can change modern browser's behaviors, such as changing the submit button for the keypad to **Search** instead of **Go** or **Enter**.
- `type="url"`: Used for web addresses. Some web browsers may provide client-side validation for the URL for this field.
- `type="tel"`: Used for telephone numbers.



For some background reading on HTML5 input types, see <http://html5doctor.com/html5-forms-input-types/>.

Copy the `register.phtml` form from `app/design/frontend/base/default/template/persistent/customer/form` to `app/design/frontend/default/responsive/template/persistent/customer/form` and open it to begin making changes. First, you are likely to find the following code:

```
<div class="input-box">
<input type="text" name="email" id="email_address" value="
 <?php echo $this->escapeHtml($this->getFormData()->getEmail())
 ?>" title="<?php echo $this->__('Email Address') ?>"
 class="input-text validate-email required-entry" />
</div>
```

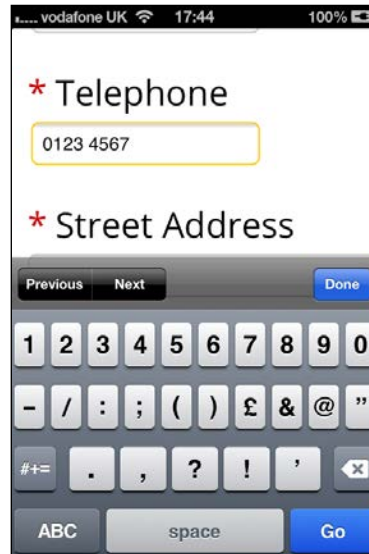
As before, change the `type` attribute in the preceding input element to `email` so that it reads as follows:

```
<div class="input-box">
<input type="email" name="email" id="email_address" value="
 <?php echo $this->escapeHtml($this->getFormData()->getEmail())
 ?>" title="<?php echo $this->__('Email Address') ?>"
 class="input-text validate-email required-entry" />
</div>
```

Further down the template file, you will find an input for the customer's telephone number, which you can change to be `type="tel"`:

```
<div class="input-box">
<input type="tel" name="telephone" id="telephone" value="
 <?php echo $this->escapeHtml($this->getFormData()->
 getTelephone()) ?>" title="<?php echo $this->__('Telephone') ?>"
 class="input-text <?php echo $this->helper('customer/address')->
 getAttributeValidationClass('telephone') ?>" />
</div>
```

At the moment, this field presents a keyboard when viewed on a smartphone, as shown in the following screenshot:



Once uploaded, you can see the change take effect for those fields you changed. If, for instance, you attempt to edit the **Telephone** field on the customer registration page, you will now be presented with a more relevant numerical keypad on most smartphone devices, as shown in the following screenshot:

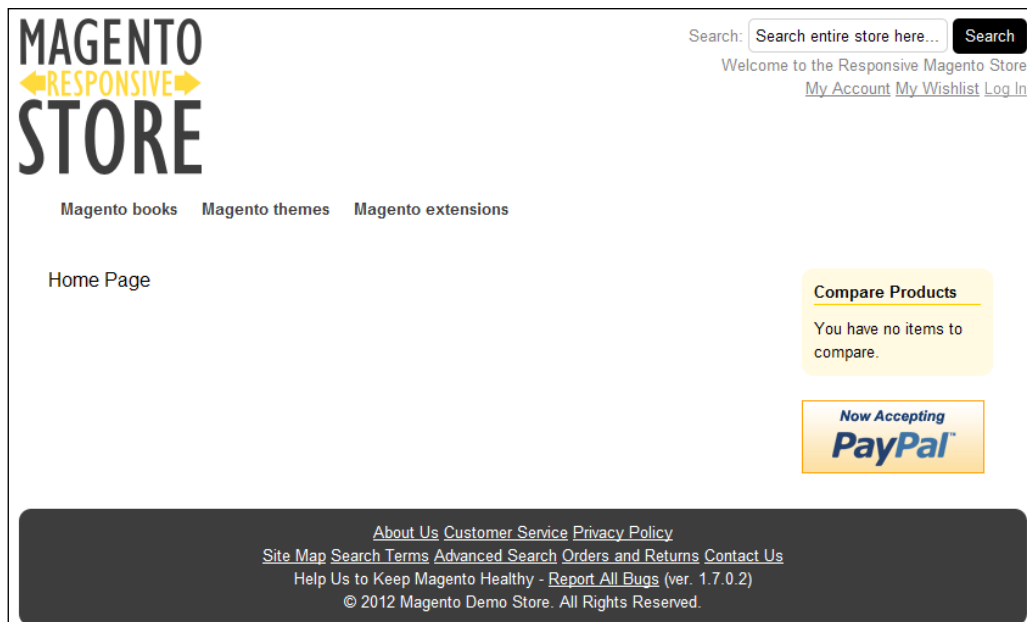


If you can't see the address or telephone number fields at the registration process, you may need to add the following code to your theme's `local.xml` file:

```
<customer_account_create>
<reference name="customer_form_register">
<action method="setShowAddressFields"><param>true</param></action>
</reference>
</customer_account_create>
```

## Using the CSS @font-face rule to use custom fonts in your Magento theme

So far, your new Magento theme makes use of standard fonts provided by the browser, as shown in the following screenshot:



By using CSS @font-face rule, it's possible to add your own custom fonts to your theme. One of the nicest ways to do this is through Google Fonts. The Google Font page (<http://www.google.com/fonts>) provides a HTML snippet to embed a URL to the stylesheet specific to the font you want to use, which will look something similar to the following code:

```
<link href='http://fonts.googleapis.com/css?family=Open+Sans'
 rel='stylesheet' type='text/css'>
```


All you require from this is the value of the href attribute – the URL to the font's stylesheet itself at `http://fonts.googleapis.com/css?family=Open+Sans`.

Once you have selected your preferred font (for example, `http://www.google.com/fonts#UsePlace:use/Collection:Open+Sans`), open your theme's `styles.css` file in the `skin/frontend/default/responsive/css` directory and place the following line at the top of your CSS file to import the new font in to your file:

```
@import url('http://fonts.googleapis.com/css?family=Open+Sans');
```

You can then use the new font in your Magento theme's stylesheet. For example, to change all fonts throughout the website, to use the new font, and add the `font-family` attribute to the body element:

```
body {
 /* Other CSS */
 font-family: "Open Sans", Arial, sans-serif;
}
```

[  Replace Open Sans in the preceding code with the name of your chosen font. ]

If you save this change and then refresh your Magento store, you will see that the font throughout has changed, as shown in the following screenshot:





You will need to include the HTTPS URL for your chosen web font to prevent SSL errors occurring in secure pages. Just replace `http://` in the preceding code with `https://` in the URL to the font file from Google Fonts.

## Summary

This chapter has helped you to enhance your Magento theme for your customers, including:

- Using the `css3-mediaqueries.js` polyfill to help Internet Explorer support media queries in older versions, so your website is functional for as many customers as possible
- Making use of the CSS `@font-face` rule to customize the look and feel of your store to customers
- Customizing Magento's form inputs to better help customers on mobile and tablet devices to enter their details

Throughout this book, you have been guided towards creating a simple responsive Magento theme. Magento is an extensive e-commerce system, so there is probably much that you still want to change and update from this theme, but with the tips and tricks here you should be off to a good start.



# Index

## A

- account log**
  - setting 73-76
- Add New Block button** 30
- advanced navigation** 24

## C

- cart tables**
  - styling 62, 64
- category grid view** 50, 51
- category list view** 46, 47
- common breakpoints** 19
- CSS**
  - adding, for changing footer link 40, 41
  - adding, for changing header link 40, 41
  - adding, for larger screens 39
- CSS @font-face rule**
  - used, to add custom fonts in Magento theme 89, 90
- CSS footer**
  - adding 38, 39
- CSS header**
  - adding 38, 39
- CSS media queries**
  - supporting, in Internet Explorer with css3-mediaqueries.js 81, 82
  - URL 81
- CSS styling**
  - adding, for Magento theme 6, 7
- custom fonts, adding**
  - CSS @font-face rule, using 89, 90

## D

- definitive style**
  - making 44, 45
- dropdown navigation**
  - for Magento store 31-36

## E

- e-commerce navigation**
  - advanced navigation 24
  - simple navigation 24
  - styling, in Magento 24, 25
- error messages**
  - styling 76

## F

- footer sitemap navigation**
  - adding 29, 30
- form elements**
  - styling 58-61

## G

- Github**
  - URL 7
- Google Font page**
  - URL 89
- Grid view** 46

## H

- header template**
  - skip-to link, adding 26-28



**height attribute**  
removing 54, 55

**href attribute** 90

**HTML5**

Magento theme doctype, changing to 84, 85

**HTML5 input types**

URL 87

## I

**images**

styling, in Magento theme 19-23

**input type value**

changing, for login 85, 86

changing, for registration 86-89

**Internet Explorer**

CSS media queries, supporting with  
css3-mediaqueries.js 81, 82

## J

**JavaScript file**

adding, to Magento theme through local.  
xml 83

## L

**List view** 46

**login**

input type value, changing for 85, 86

## M

**Magento**

e-commerce navigation, styling 24, 25

Grid view 46

List view 46

Responsive one page checkout, using 68-72

Responsive product page layout 42

Responsive shopping cart, using 57

**Magento cart page**

shipping estimate tool, removing from 62

**Magento customer account pages**

account log, setting 73-76

error messages, styling 76

My Applications, removing from customer  
account 78

My Applications, removing from Magento  
customer account area 79

My Downloadable Products links,  
removing from customer account 78,  
79

styling 73

**Magento inheritance system**

working, URL 10

**Magento search results**

dealing with 52

product images, resizing 53

**Magento store**

dropdown navigation, using for 31-36

skip to footer navigation, adding 25

**Magento store data entry**

improving, for customers 84

**Magento templates**

overwriting 14, 15

**Magento theme**

basic CSS styling, adding for 6, 7

basic XML layout, adding for 7, 8

creating 5, 6

default Magento templates,

overwriting 14, 15

enabling 11-13

hierarchy 9, 10

images, styling 19-23

JavaScript file, adding through local.xml 83

media queries, adding 15-19

meta viewport element, adding 8, 9

**Magento theme doctype**

changing, to HTML5 84, 85

**Magento theme hierarchy**

working, URL 10

**Magento theme hierarchy and fallbacks**

URL 10

**max-width property** 53

**media queries**

adding, to Magento theme 15-19

**meta viewport element**

adding 8, 9

**Mozilla's Developer**

URL 15

**My Applications**

removing, from customer account 78, 79

**My Downloadable Products links**  
removing, from customer account 78, 79

## N

**native browser support**  
of media queries, URL 55

## P

**pagination option** 48, 49  
**product image**  
laying out 43  
resizing 53  
**product information**  
laying out 43  
**product pager option** 48, 49

## R

**registration**  
input type value, changing for 86-89  
**Responsive category page layout**  
about 46  
category grid view 50, 51  
category list view 46, 47  
pagination option 48, 49  
product pager option 48, 49  
sort by dropdown option 48, 49  
**Responsive one page checkout**  
in Magento 68-72  
**Responsive product page layout**  
definitive style, making 44, 45  
in Magento 42  
product image, laying out 43  
product information, laying out 43  
**Responsive shopping cart**  
cart tables, styling 62, 64  
form elements, styling 58-61  
in Magento 57  
shipping estimate tool, removing from  
Magento cart page 62

unnecessary table columns, removing for  
mobile 64-67

## S

**Save Block button** 30  
**Save Config button** 13  
**Save Page button** 22  
**shipping estimate tool**  
removing, from Magento cart page 62  
**Show / Hide Editor button** 21  
**simple navigation** 24  
**skip to footer navigation**  
adding, to Magento store 25  
**skip-to link**  
adding, in header template 26-28  
**Skip to navigation button** 30  
**sort by dropdown option** 48, 49

## T

**type attribute** 86, 87

## U

**unnecessary table columns**  
removing, for mobile 64-67  
removing, for smaller screened  
devices 64-67

## W

**website footer**  
laying out 37, 38  
**website header**  
laying out 37, 38  
**width attribute**  
removing 54, 55

## X

**XML layout**  
adding, for Magento theme 7, 8





## Thank you for buying **Magento Responsive Theme Design**

### **About Packt Publishing**

Packt, pronounced 'packed', published its first book "*Mastering phpMyAdmin for Effective MySQL Management*" in April 2004 and subsequently continued to specialize in publishing highly focused books on specific technologies and solutions.

Our books and publications share the experiences of your fellow IT professionals in adapting and customizing today's systems, applications, and frameworks. Our solution based books give you the knowledge and power to customize the software and technologies you're using to get the job done. Packt books are more specific and less general than the IT books you have seen in the past. Our unique business model allows us to bring you more focused information, giving you more of what you need to know, and less of what you don't.

Packt is a modern, yet unique publishing company, which focuses on producing quality, cutting-edge books for communities of developers, administrators, and newbies alike. For more information, please visit our website: [www.packtpub.com](http://www.packtpub.com).

### **About Packt Open Source**

In 2010, Packt launched two new brands, Packt Open Source and Packt Enterprise, in order to continue its focus on specialization. This book is part of the Packt Open Source brand, home to books published on software built around Open Source licences, and offering information to anybody from advanced developers to budding web designers. The Open Source brand also runs Packt's Open Source Royalty Scheme, by which Packt gives a royalty to each Open Source project about whose software a book is sold.

### **Writing for Packt**

We welcome all inquiries from people who are interested in authoring. Book proposals should be sent to [author@packtpub.com](mailto:author@packtpub.com). If your book idea is still at an early stage and you would like to discuss it first before writing a formal book proposal, contact us; one of our commissioning editors will get in touch with you.

We're not just looking for published authors; if you have strong technical skills but no writing experience, our experienced editors can help you develop a writing career, or simply get some additional reward for your expertise.



## Instant E-Commerce with Magento: Build a Shop

ISBN: 978-1-78216-486-9

Paperback: 52 pages

A fast-paced, practical guide to building your own shop with Magento

1. Learn something new in an Instant! A short, fast, focused guide delivering immediate results
2. Learn how to install and configure an online shop with Magento
3. Tackle difficult tasks like payment gateways, shipping options, and custom theming
4. Full of clear screenshots and step-by-step instructions



## Magento 1.3: PHP Developer's Guide

ISBN: 978-1-84719-742-9

Paperback: 260 pages

Design, develop, and deploy feature-rich Magento online stores with PHP coding

1. Extend and customize the Magento e-commerce system using PHP code
2. Set up your own data profile to import or export data in Magento
3. Build applications that interface with the customer, product, and order data using Magento's Core API
4. Packed with examples for effective Magento Development

Please check [www.PacktPub.com](http://www.PacktPub.com) for information on our titles



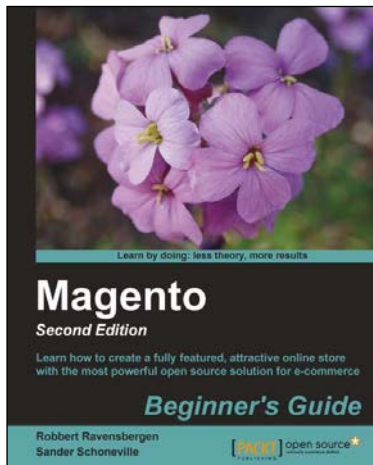
## Instant Magento Shipping How-to

ISBN: 978-1-78216-540-8

Paperback: 58 pages

Making Magento shipping settings work for your business

1. Learn something new in an Instant! A short, fast, focused guide delivering immediate results
2. Set up popular shipping methods such as Table Rates
3. Easily manage orders with reports and tools
4. Order through the shipment phase



## Magento Beginner's Guide - Second Edition

ISBN: 978-1-78216-270-4

Paperback: 320 pages

Learn how to create a fully featured, attractive online store with the most powerful open source solution for e-commerce

1. Install, configure, and manage your own e-commerce store
2. Extend and customize your store to reflect your brand and personality
3. Handle tax, shipping, and custom orders

Please check [www.PacktPub.com](http://www.PacktPub.com) for information on our titles