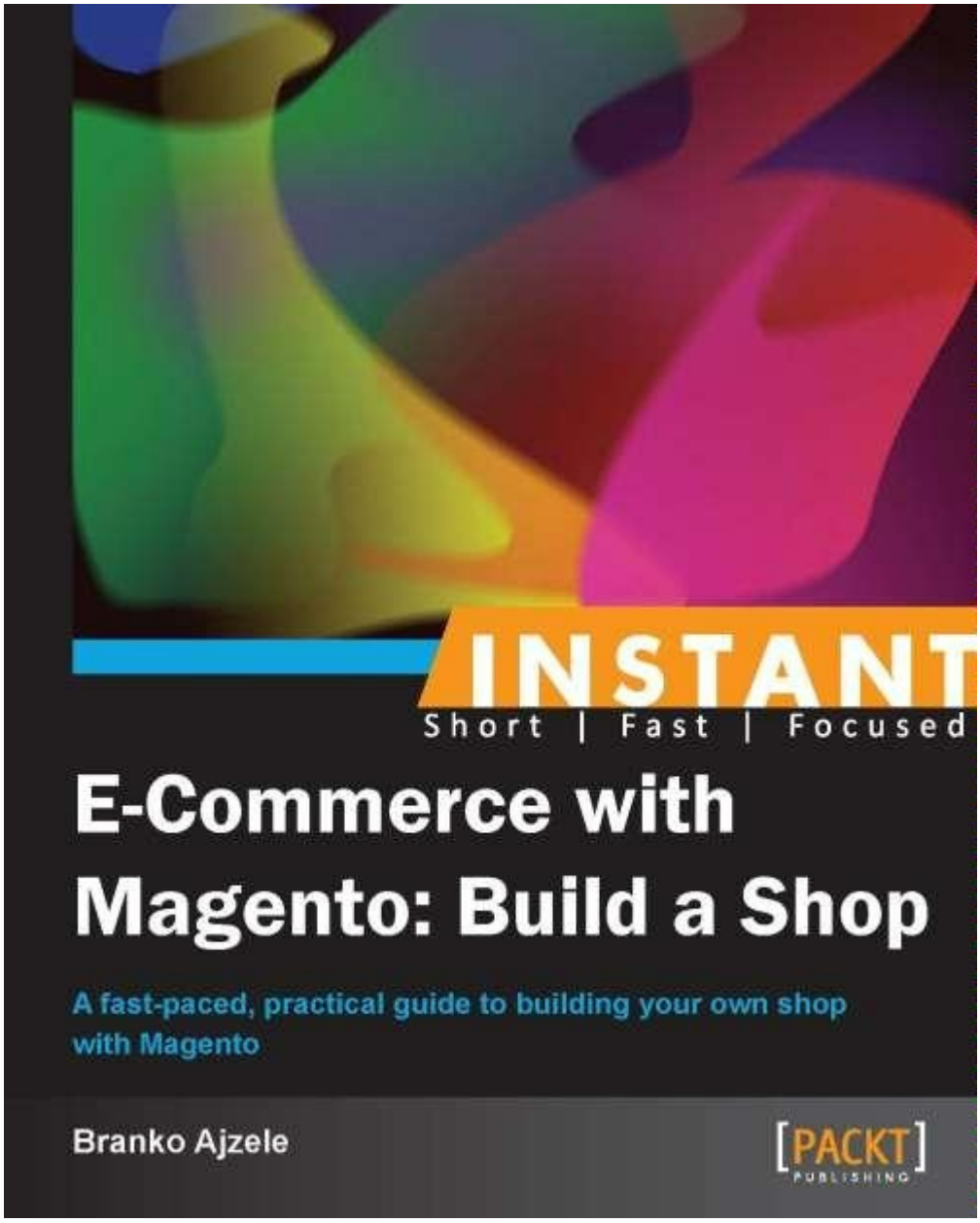

INSTANT
Short | Fast | Focused
E-Commerce with Magento: Build a Shop
A fast-paced, practical guide to building your own shop with Magento
Branko Ajzele
[PACKT]
PUBLISHING

Table of Contents

Instant E-Commerce with Magento: Build a Shop
Credits
About the Author
About the Reviewer
www.PacktPub.com
    Support files, eBooks, discount offers and more
        Why Subscribe?
        Free Access for Packt account holders
Preface
    What this book covers
    What you need for this book
    Who this book is for
    Conventions
    Reader feedback
    Customer support
        Errata
        Piracy
        Questions
1. Instant E-Commerce with Magento: Build a Shop
    Installation (Should know)
        Getting ready
        How to do it...
        How it works...
        There's more...
    Basic configuration (Must know)
        Getting ready
        How to do it...
        How it works...
        There's more...
    Categories (Must know)
        Getting ready
        How to do it...
        How it works...
    Product attributes and attribute sets (Must know)
        Getting ready
        How to do it...
        How it works...
        There's more...
    Products (Must know)
        Getting ready
        How to do it...
        How it works...
        There's more...
    Payment and shipping method (Must know)

- How to do it...
- How it works...

Transactional e-mails (Must know)
- How to do it…
- How it works...

Catalog and Shopping Cart Price Rules (Must know)
- Getting ready
- How to do it...

Managing orders (Must know)
- How to do it...
- How it works…

Themes and appearance (Should know)
- Getting ready
- How to do it...

Instant E-Commerce with Magento: Build a Shop

Instant E-Commerce with Magento: Build a Shop

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: February 2013

Production Reference: 1180213

Published by Packt Publishing Ltd.

Livery Place

35 Livery Street

Birmingham B3 2PB, UK.

ISBN 978-1-78216-486-9

www.packtpub.com

Credits

Author

Branko Ajzele

Reviewer

Matej Krivak

Acquisition Editor

Andrew Duckworth

Commissioning Editor

Ameya Sawant

Technical Editor

Varun Pius Rodrigues

Project Coordinator

Sherin Padayatty

Proofreader

Aaron Nash

Production Coordinators

Conidon Miranda

Prachali Bhiwandkar

Cover Work

Conidon Miranda

Prachali Bhiwandkar

About the Author

Branko Ajzele was born on January 16th, 1983 in Osijek, Croatia. His high school and faculty education was in the field of electro-technology. He graduated at Faculty of Electrical Engineering, in Osijek in 2006. Although not directly related to IT, during his faculty days he fell in love with web-related technologies. Throughout his career he initially worked full time as a Computer Consultant and later as a full time IT System Administrator. After that he got hooked on programming. He has spent over six years learning and developing both desktop and web applications in different programming environments such as C#, Java, and later in PHP.

He is currently working as CTO / Magento Certified Developer Plus at a company called Inchoo. Inchoo specializes in building and delivering high quality e-commerce solutions mostly powered by Magento eCommerce platform. At Inchoo he gained over four years of everyday Magento development and e-commerce project management experience. Additionally he is highly experienced with Zend Framework, Doctrine ORM, Symfony, and PHP in general. Besides web technologies, his latest interests are mobile platforms such as Android and iOS.

About the Reviewer

Matej Krivak is an experienced web/database developer, born and raised in Osijek, Croatia.

He has a Master's degree in Engineering (more specifically, Process Computing) from the Faculty of Electrical Engineering in Osijek, where he graduated with highest honors (that is, summa cum laude).

Matej is currently working for Inchoo, a Croatian company specialized in creation of e-commerce solutions based on Magento, as a team leader and a senior back-end developer. In the past, he has worked for Dialog, a small Croatian company, where he developed and maintained systems based on Oracle technologies.

He has a good knowledge of Magento, Oracle Designer 9i/11g, and various relational database management systems such as MySQL, Oracle, and Microsoft SQL Server.

www.PacktPub.com

Support files, eBooks, discount offers and more

You might want to visit www.PacktPub.com for 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.

Preface

So you've decided to build and run your own web shop, but you need some guidance about where to start? You have come to the right place. May we suggest the Magento eCommerce platform. Magento comes in various editions: Community, Enterprise, and GO. Each edition has its strengths and weaknesses. Recipes contained in this micro book are aimed at Magento Community Edition (CE).

Magento CE is free, open source software that enables developers and tech savvy merchants to explore the power and flexibility of the Magento platform. In terms of platform all three product editions (Community, Enterprise, GO) share the same code base and functionality so getting to know Magento CE means a step closer to getting to know the rest of the product line.

Magento platform is built on top of Zend Framework, which implies PHP programming language. It uses MySQL database for data storage, and an HTTP server like Apache for serving content as web pages. All of these technologies can be run on Unix-based operating systems like Linux, OSX, BSD or even on Microsoft Windows. This makes Magento a pretty portable piece of software. Most of the time, however, this is not something you will trouble yourself with.

In order to successfully work through all of the recipes in this book, a low cost third-party shared hosting service will be fine. However, once you get the hang of Magento, you should carefully choose your hosting provider, or host Magento yourself if you have the know-how.

Recipes in this book cover everything you need to get your Magento platform up and running.

What this book covers

Installation (Should know) walks you through and basic Magento installation process, from obtaining the installation files, to the final installation success screen.

Basic configuration (Must know) walks you through the basic required, and some often forgotten, post-installation configuration steps.

Categories (Must know) shows you how to manage categories, which are the basic structure building blocks of your web shop.

Product attributes and attribute sets (Must know) emphasizes the importance of knowing the role of product attributes, which are the basic

building blocks of each product. This recipe guides you though management of both attributes and their sets.

Products (Must know) shows you how to manage products, which are the basic content building blocks of your web shop.

Payment and shipping method (Must know) gives an overview of out-of-the-box available payment and shipping methods and their configuration.

Transactional e-mails (Must know) outlines the importance and configuration of this often forgotten feature of Magento.

Catalog and Shopping Cart Price Rules (Must know) guides you through creating rules for your store by providing two practical examples of rules.

Managing orders (Must know) is a sort of backbone to this book, providing instructions on order management, which is integral and probably the most important part of running your store.

Themes and appearance (Should know) is sort of the sugar on top, giving a short description and guide on how to visually improve your store appearance.

What you need for this book

In order to successfully go through all of the recipes mentioned in this book you need to have your web server environment set up. This implies having PHP, MySQL, and Apache HTTP server configured and running. If you are not a developer, don’t worry. This is not something that you need to do by yourself; you can simply look for third-party hosting providers. You can check out http://www.magentocommerce.com/partners/hosting/ for a list of specialized Magento hosting providers.

Who this book is for

This book is primarily intended for merchants starting their own new shop and developers completely new to the Magento eCommerce platform.

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 are shown as follows: “ Template variable calls are done like `{{var order.increment_id}}`.”

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: “ Read the terms and conditions and click on the **Continue** button.”

> ### Note
>
> Warnings or important notes appear in a box like this.

> ### Tip
>
> 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.

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.

Chapter 1. Instant E-Commerce with Magento: Build a Shop

Welcome to *Instant E-Commerce with Magento: Build a Shop*. Over the last few years Magento has positioned itself as one of the leading open source e-commerce platforms. A number of customer and merchant-friendly features are to thank for that. The fact that it's built on top of PHP/MySQL made it really accessible to a large community of web developers, thus circling the merchant - customer - developer cycle to everyone's satisfaction.

This book is primarily targeted towards merchants and developers new to the Magento platform, or new to the entire concept of running a web shop. It will provide you with useful information on how to set up, configure, and run your Magento-powered web shop.

Installation (Should know)

This installation guide will be focused towards **Magento Community Edition** (later renamed as **Magento CE**).

Getting ready

Prior to installation, we need to make sure that our web server system requirements are met. You can find the official list of system requirements at http://www.magentocommerce.com/system-requirements.

If you are not that much familiar with the server details, you can just download the `magento-check.php` file from the http://www.magentocommerce.com/knowledge-base/entry/how-do-i-know-if-my-server-is-compatible-with-magento page, put it onto the web server in the publicly accessible directory, and then navigate your browser to it. It will tell you if the server requirements are met or not.

Additionally, you need to have MySQL database created, with UTF-8 encoding. If you are not sure how to do this, contact your hosting support to do it for you and send you the access data (username/password, host, and so on). You will then use this database access data during installation process.

How to do it...

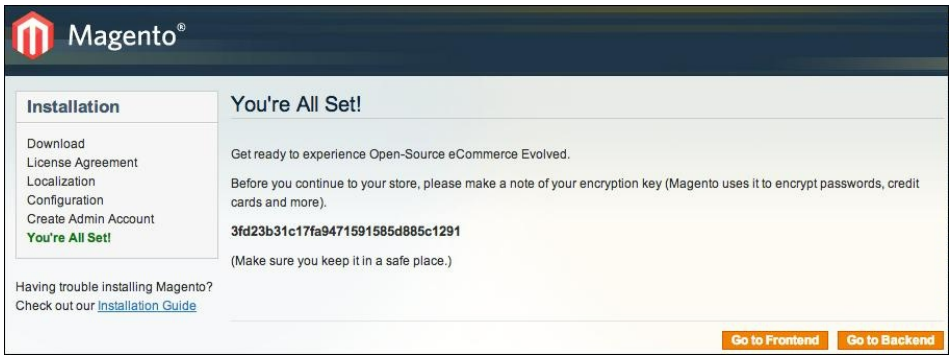
1. Download a copy of the Magento CE installation archive from http://www.magentocommerce.com/download. Grab the one under the **Full Release** section (select any format you like).
2. Unpack the Magento installation archive under your server root folder, root being the folder that will be targeted when you point your browser to your shop root URL, for example, http://myshop.com.
3. Now point your browser to your shop URL and if all is fine you should be able to see a Magento installation screen titled **Welcome to Magento's Installation Wizard!** as shown in the following screenshot:



4. This is the first out of the five installation steps. Read the terms and conditions, then check **I agree to the above terms and conditions** and click on the **Continue** button.
5. You should now be able to see a second installation screen called **Localization**. Here we set **Locale**, **Time Zone,** and **Default Currency**. Set the **Locale** options to suit your needs and click on the **Continue** button.
6. You should now be able to see a third installation screen called **Configuration**. Configuration is divided into three sections: **Database Connection**, **Web access options**, and **Session storage options**. Usually you just need to configure the **Database Connection** section and leave the other two at their default values, unless you know exactly what you are doing. Once you configure **Database Connection,** click on the **Continue** button. Please note that jumping to the next step can take some time, up to minutes if your web and database server are slow. Just leave your browser open and wait until you are redirected to next step.
7. You should now be able to see fourth installation screen called **Create Admin Account**. This screen is also divided into several sections: **Personal Information**, **Login Information**, and **Encryption Key**. Since this is a new installation, you just need to

configure the **Personal Information** and **Login Information** sections, leaving the **Encryption Key** section at default, which is blank. Once you configure the required sections click on the **Continue** button.

8. You should now be able to see the last, fifth installation screen called **You're All Set!** as shown in the following screenshot:



> **Note**
>
> As suggested by the instructions in the preceding screenshot, please make a note of the encryption key shown on the page. The encryption key is used by Magento to encrypt the sensitive configuration data.

How it works...

The installation process can take up to a couple of minutes, depending on your server parameters. By default you can access your web shop frontend interface at a URL like http://myshop.com and backend administration interface at a URL like http://myshop.com/index.php/admin.

There's more...

It is also possible to install sample data so that after installation your store contains some products and categories. This is useful for merchants and developers who want to spare some time on testing Magento. You can download the official Magento sample data from the http://www.magentocommerce.com/download page, under the **Sample Data** section. It is important to know that sample data is imported into MySQL database before installation. You cannot import sample data into the database after you have already installed Magento.

Basic configuration (Must know)

This guide describes the basics required and the often forgotten configuration information for your web shop.

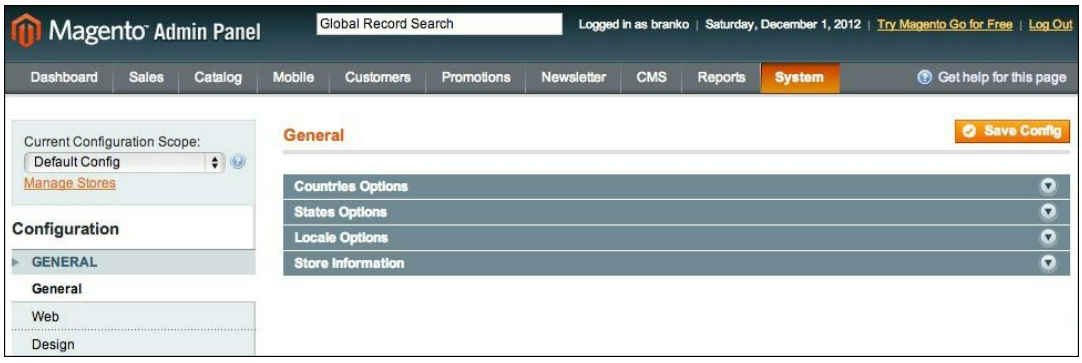
Getting ready

Unless you specified a custom path in the **Admin Path** option under the **Configuration** screen of the installation process, you can access your Magento backend administration through a URL like http://myshop.com/index.php/admin. If you specified a custom value under **Admin Path** during installation then you should be able to access your store through a URL like http://myshop.com/index.php/myCustomValue.

Once you trigger the backend administration URL, you should see the following screen (if you are not already logged in to admin):



Once you access the admin area select **System** | **Configuration** from the top menu, this should get you to a configuration screen as shown in following screenshot:



Although Magento supports robust store hierarchy with one or more websites, each with one or more store views, we will focus on the basic *single website + single store view* configuration. This is the configuration you get after the installation. You are perfectly fine leaving the **Current**

Configuration Scope on the **Default Config** value as shown in following screenshot:



Even though your website would probably work without any options changed, it is absolutely essential that after the installation and prior to going live with your website you configure the following sections (including their sub-sections):

- **General** | **General**
- **General** | **Web**
- **General** | **Design**
- **General** | **Currency Setup**
- **General** | **Store Email Addresses**
- **General** | **Contacts**
- **Customers** | **Customer Configuration**
- **Sales** | **Shipping Methods**
- **Sales** | **Payment Methods**

Leaving these values at their default will make you look irresponsible as a merchant and might affect your customers' behavior. For example, you do not want your customers to get successful registration e-mails titled **Welcome to Magento Commerce**.

How to do it...

1. Log in to Magento backend administration and navigate to **System** | **Configuration**.
2. On the left-hand side, make sure **Current Configuration Scope** is set to **Default Config**.
3. On the left-hand side, under **General**, locate and click on the **General** tab.
4. Click on the **Countries Options** section; this should open several configuration options as shown in the following screenshot:



5. Set the proper value of each configuration option and click on the **Save Config** button.
6. Now repeat the Steps 3 to 5 for at least every configuration section mentioned previously in the recipe as essential configuration.

Unless you won't be running a multi-website from the start, we suggest setting **Share Customer Accounts** option in **Customers** | **Customer Configuration** to **Global**. This means that even if you add multiple websites later, your customer will able to log in to any of those websites.

> ### Note
>
> We advise that in the early days of your live production website you set the **Enabled** option in **Advanced** | **Developer** | **Log Settings** to **Yes**. This will generate any possible Magento system logs under the `/var/logs/` folder and help you spot the issues on time.

How it works...

Since Magento platform supports having multiple websites, with each website having multiple stores it is safe to assume that most of these configuration options can be adjusted on a website or store level. This is where the previously mentioned **Current Configuration Scope** comes in.

For example, if you had several websites defined in Magento, you can easily set the value of let's say **System** | **General** | **Design** | **Html Head** | **Default Title** for each website differently just by changing the **Current Configuration Scope** and saving the configuration options for each scope.

There's more...

Sometimes, if you change a certain configuration option and do not see the expected effects, it might be useful to clean up a cache. You can do so by going to **System** | **Cache Management** and clicking on the **Flush Magento Cache** button. There is also a **Flush Cache Storage** button there; however

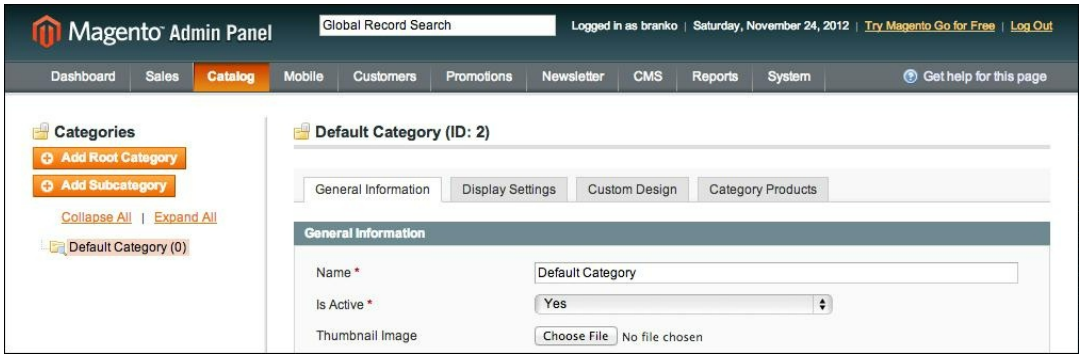
you don't want to use this one extensively on a live production site as it deletes all customer sessions when clicked.

Categories (Must know)

Categories are essential parts of the structure of your store. This recipe shows you how to create, edit, and display categories.

Getting ready

To access the category management page, log in to **Magento Admin Panel** then go to **Catalog** | **Manage Categories**, you should see a screen like the one shown in following screenshot:



Before you start creating new categories, there are several things you need to know about them.

Magento has this concept of root category. **Root category** cannot be deleted and is not visible anywhere on the frontend. It is linked to an individual store. Every category you create must be a subcategory of some root category.

How to do it...

Creating a category in Magento is pretty straightforward:

1. Log in to **Magento Admin Panel** then go to **Catalog** | **Manage Categories**.
2. Click on the **Default Category** option on the left-hand side, and then click on the **Add Subcategory** button also on the left-hand side. You should see the New Subcategory edit screen now.
3. Staying on the **General Information** tab, write down `Phones` under **Name**, set **Is Active** to **Yes**, write some text under **Description**, upload an image under **Image,** and finally set **Include** in **Navigation Menu** to **Yes** and click on the **Save Category** button on the right-hand side. If all is good you should see a message like **The category has been saved**. Even though only **Name**, **Is Active**, and **Include** in the Navigation Menu are required fields while creating a new category, it is advisable to set the **Image** and

Description as well, as these can improve the overall professional feel of your store and appeal to customers.

> ### Note
>
> Some of the input fields like **Description** have a button titled **WYSIWYG Editor** next to them. If you are not that much into writing pure HTML you can use this button and edit the text in visual manner.

4. Once you save the new category, it is always a good idea to jump to frontend to see if your new category is visible in navigation menu.
5. Creating subcategories is basically the same; just repeat steps 2 to 4. The important thing is that you first click in the category tree for which category you wish to create a subcategory.

Deleting a category in Magento is pretty straightforward:

1. Log in to **Magento Admin Panel** then go to **Catalog** | **Manage Categories**.
2. In the category tree find the category you wish to delete and click on it.
3. Now click the **Delete Category** button in the upper-right corner. Keep in mind that if the category has child categories, those will be deleted as well.

Rearranging categories in Magento is also pretty easy since the Magento category tree is drag-and-drop enabled:

1. Log in to **Magento Admin Panel** then go to **Catalog** | **Manage Categories**.
2. In the category tree find the category you wish to rearrange (move to another category), click on it with your mouse, and drag it to another category. You don't even have to click on the **Save Category** button, since Magento automatically saves drag-and-drop operations.

How it works...

If we go back to admin, and click on our newly created **Phones** category, we will notice that there are far more options we can set on the category. These options are scattered across several tabs: **General Information**, **Display Settings**, **Custom Settings**, and **Category Products,** as shown in following screenshot:



Most of the time, the **General Information** tab will have you covered. However, if you want some more advanced stuff then you need to look at other tabs for more options.

The **General Information** tab is rather self-explanatory; it contains the basic information for your category.

The **Display Settings** tab holds the options that affect the presentation or if you prefer visualization of your category to the customer on the frontend. You can achieve really stunning results here if you carefully combine **Display Mode** with **CMS Block** options. On top of that, this tab holds one more option which should be emphasized as the holy grail of category options, the **Is Anchor** option.

The **Display Settings' Is Anchor** option name is completely misleading, in regards to its functionality. Anchor is usually interpreted as a link in the web world, so one might think that this option simply stands for *this category is outputted as a link or just text*, but it's not. A more proper name for the **Is Anchor** option would be *Use Layered Navigation* because this option specifies the use of Magento's layered navigation feature.

Note

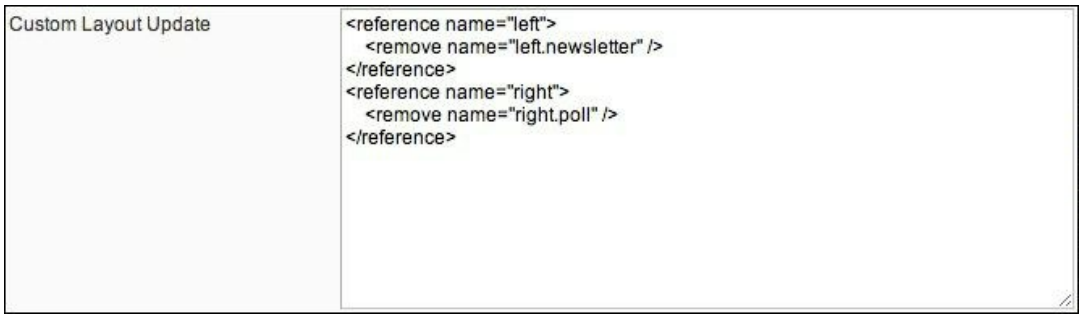
You can create various visually stunning promo materials under **CMS** | **Static Blocks**, possibly some HTML and JavaScript sliders, and so on. Then tie those to the individual category through the combination of **Display Mode** with **CMS Block** options. These static blocks would then be output on the frontend in the same manner as a category image is now.

What is **Layered Navigation**? It is a feature of Magento that enables you to provide customers with the option of searching for products through a variety of filters, such as price, color, or any other existing or additionally created product attribute. If you set **Is Anchor** of a certain category to **Yes** then subcategories of that category are displayed as layered navigation filters among other product attribute filters just mentioned, as shown in the following screenshot:



Individual product attributes have the option **Use In Layered Navigation**, where you may select **Yes** or **No**. If you plan on relying on these options then I suggest you set the category's **Is Anchor** to **Yes** as well, as it can improve the customer frontend experience.

The **Custom Design** tab provides additional set of options that affect the theme and layout used for the individual category. Depending on the settings they can also affect the child categories. In practice, this set of options is rarely used; however it should by no means be ignored. With these options you can achieve a radically different look and feel on individual categories.



The **Category Products** tab enables you to assign one or more products to this category.

Usually the merchant workflow is such that products are assigned to a category through the product edit page, and not this category edit page.

Product attributes and attribute sets (Must know)

Product attributes are one of those things where Magento really shines in the context of extensibility. Attribute is basically a property of a product, for example, name, SKU, description, weight, price, and so on.

What's great about attributes is that they enable you to add new properties to your products without the need to write any new code. It is all handled through Magento backend administration. This way, merchants can easily manage the product attributes without developers' help.

Getting ready

Before you start creating your own attributes and attribute sets, it is important to understand a theoretical background of the two.

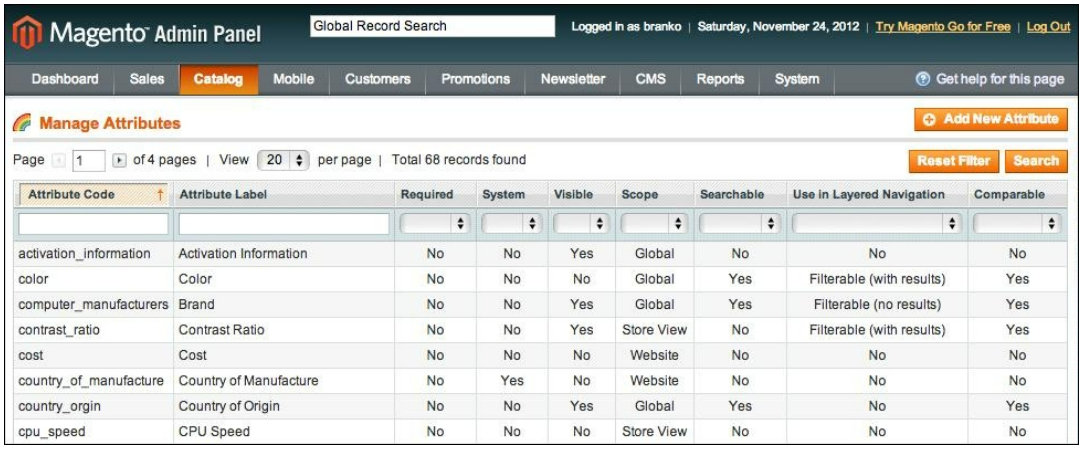
Why do we even need attributes? Well, imagine that you had a shop that came with a predefined set of attributes, let's say name, SKU, description, weight, and price. Further imagine you will be selling shoes. Obviously you lack at least one attribute, the shoe size for the customer to select.

Instead of just asking the developer for assistance, in Magento you simply create a new attribute.

What about attribute sets? Attribute sets are sort of containers for individual group of attributes. Imagine your shop sells phones and shoes, not a likely scenario but play along. Obviously phones and shoes would have some attributes in common but many different. Common attributes might be name, SKU, description, weight, price, and so on. Then a phone might have attributes like screen size, battery life, operating system, while shoes might have attributes like shoe size, material, and so on.

This is where attribute sets come in. In this case we would create two attribute sets, one called `Shoes` and another called `Phones`. In both attributes sets we would throw in common attributes like name, SKU, description, weight, price, then depending on the *phone versus shoe* debate we would assign the rest of the attributes to proper attribute sets.

Managing Product Attributes is done from Magento backend administration under the **Catalog** | **Attributes** | **Manage Attributes** page as shown in the following screenshot:



Attribute Code	Attribute Label	Required	System	Visible	Scope	Searchable	Use in Layered Navigation	Comparable
activation_information	Activation Information	No	No	Yes	Global	No	No	No
color	Color	No	No	No	Global	Yes	Filterable (with results)	Yes
computer_manufacturers	Brand	No	No	Yes	Global	Yes	Filterable (no results)	Yes
contrast_ratio	Contrast Ratio	No	No	Yes	Store View	No	Filterable (with results)	Yes
cost	Cost	No	No	No	Website	No	No	No
country_of_manufacture	Country of Manufacture	No	Yes	No	Website	No	No	No
country_orgin	Country of Origin	No	No	Yes	Global	Yes	No	Yes
cpu_speed	CPU Speed	No	No	No	Store View	No	No	No

If you look carefully at the grid in the preceding screenshot you can see a subset of options that each attribute holds. There are options like **Required**, **System**, **Visible**, **Scope**, **Searchable**, and many more on the actual attribute edit page.

Quick clarification on some of those options:

- **Required**: This means that you need to provide a value for this attribute during product creation.
- **System**: These are the attributes that you get right out of the box when you install Magento, and without these attributes the system would not work. These are the attributes that are common to most products (phones, shoes, and many more).
- **Scope**: Since Magento is multi-website, multi-store enabled platform, you can even go that far to say that individual attribute is only visible on a website or store level. This guide works with single website single store setup so for us, this value is by default set to **Store View** so it's fine to leave it as such for all of our newly-created attributes.
- **Searchable**: This tells Magento if this attribute is searchable through its advanced search.

How to do it...

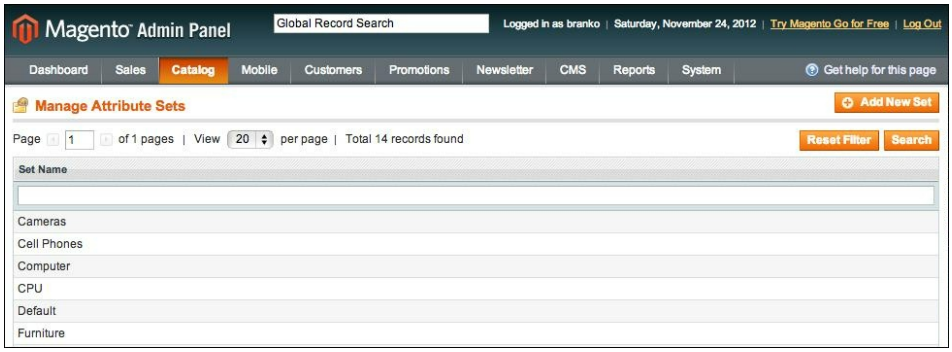
Creating a new attribute is pretty straightforward; here is what you need to do:

1. Log in to **Magento Admin Panel** then go to **Catalog** | **Attributes** | **Manage Attributes**.
2. Click on the **Add New Attribute** button in the upper-right corner. This should take us to the **New Product Attribute** page.
3. For **Attribute Code** use `free_gift_included`, for **Apply To** select **All Product Types**, for the **Admin** field in the **Manage Titles**

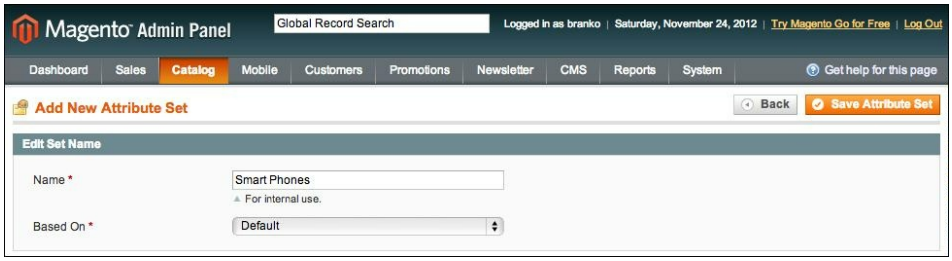
section use **Free Gift Included**, and then click on the **Save** button in the upper-right corner. Other options are not mandatory.

Now let's create a new attribute set:

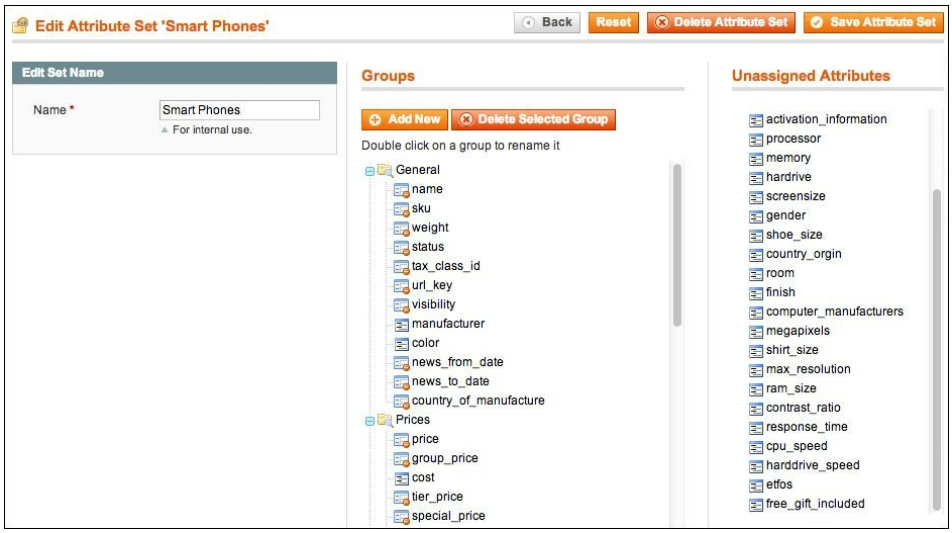
1. Go to the **Catalog** | **Attributes** | **Manage Attributes Sets** page, you should see a screen similar to the following screenshot, possibly with just the **Default** attribute set listed in the grid:



2. Click on the **Add new Set** button in the upper-right corner. This should get you to the first of two **Add New Attribute Set** screens like that shown in following screenshot. Write down `Smart Phones` under **Name** and choose **Default** under **Based On**. As the option name hints at, new attribute sets you create are based on existing ones. When you install Magento, there is one pre-existing attribute set called **Default** so all your new attribute sets will initially have to be based on that one.



3. Once you enter the required information in the preceding screenshot, click on the **Save Attribute Set** button in the upper-right corner. This will get you to the **Edit Attribute Set 'Smart Phones'** screen as shown in following screenshot:



4. Now simply click and drag our **free_gift_included** attribute from the **Unassigned Attributes** column to the **General** folder under the **Groups** column, and click on **Save Attribute Set** when you are done.

How it works...

When creating the attributes you might see a system message saying **One or more of the Indexes are not up to date: Catalog Search Index. Click here to go to Index Management and rebuild required indexes**. If you do, don't panic. This is perfectly normal; once we are done with our attribute editing you can just jump to **System** | **Index Management** and re-index everything that needs to be re-indexed.

Since we clicked on the **Save and Continue Edit** button Magento saved the attribute but left us on the attribute edit page. Once you save the attribute you cannot change its **Attribute Code** value, so be very careful what you use for the **Attribute Code** value, because if you don't like it, you will have to delete the entire attribute and create a new one.

The way we saved the attribute in the preceding example, it won't be visible on the frontend on product view page or in search. It's basically just here as a storage container on the product but it's not visible to the customer. In order to make it visible and searchable, please set the following properties in the **Frontend Properties** section in the **Properties** tab to **Yes**:

- **Use in Quick Search**
- **Use in Advanced Search**
- **Visible on Product View Page on Frontend**

Once you set those up, click on **Save Attribute** button.

There's more...

Creating attributes and attribute sets is usually a largely underestimated step often overlooked by merchants who are eager to get their shop selling, which is not good as product attribute sets are the foundations upon which each product is built.

Products (Must know)

Magento CE out of the box supports several types of products, each with its own specific functionality defined through code. Be sure to familiarize yourself with each product type prior to entering large amounts of products in your shop.

Getting ready

Get to know all the Magento product types, which are as follows:

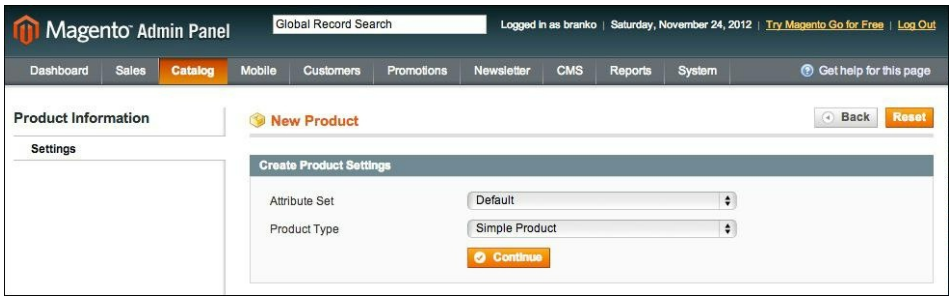
- **Simple product**: This product is the most basic product type, and the one that is mostly used. Suits all kind of physical products that require shipping, and much more.
- **Configurable product**: This product is comprised of several associated products, each representing a variation of options defined through attributes. For example, T-shirts in two colors and three sizes would account for single configurable product with six variations, that is six associated products each representing a single variation. It's like having a single Simple product, let's say Green color and L size assigned to a configurable product as a single option for the customer to select on the frontend. The good thing about Configurable products is that Magento tracks the inventory for each variation.
- **Grouped product**: This product is like a container for several simple products. Once you add a Grouped product to the cart it ends up in the cart as several individual products and not one grouped.
- **Bundle product**: This product can be partially interpreted as something between configurable and grouped, although that would be an unfair comparison. You can think of a Bundle product as a product that is composed of components that are presented in different ways and are made from existing products. Think custom-build PCs where you can configure how much RAM or what kind of monitor is included with the computer, and so on.
- **Virtual product**: This product is almost like Simple product, but it lacks the system *weight* attribute, which further means it has no shipping actions associated to it. This type of product is usually used for warranties and services.
- **Downloadable product**: This product is similar to Virtual in terms of the *weight* attribute as it has none, so shipping does not apply. Additionally it has an entire set of *downloadable files* functionality coded around it. It's used to represent various digitally downloadable content such as image, PDF files, application files, and so on.

Now that we have defined the existing product types, we can proceed with product management. This guide will cover only the Simple product type.

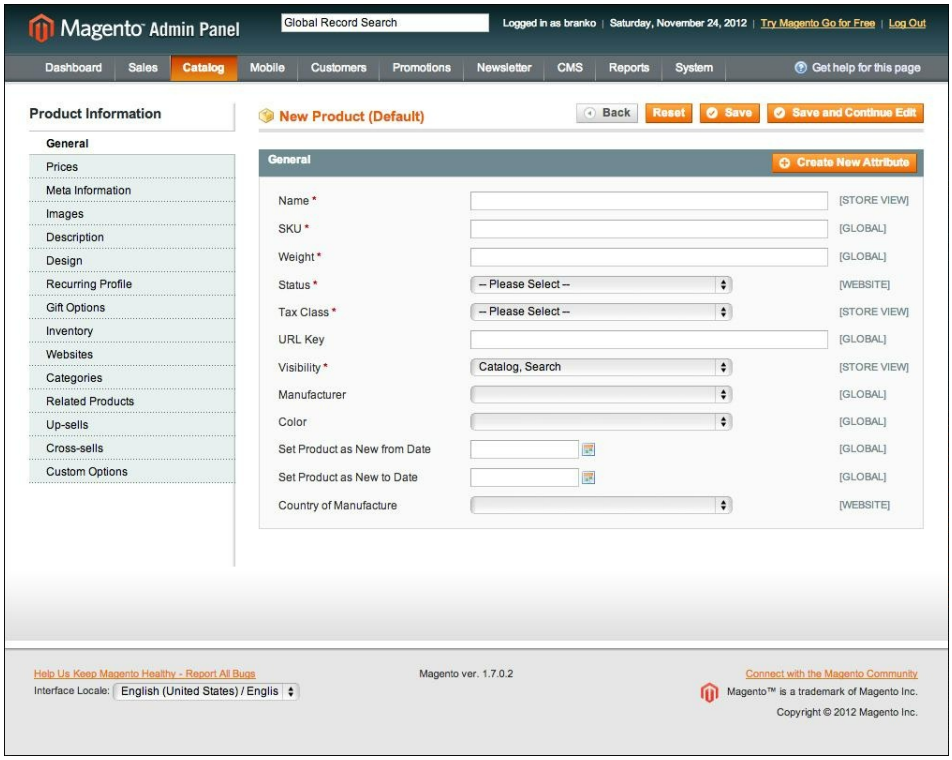
Please be sure that you have first planned and defined proper attributes and attribute sets. Only attribute sets can be assigned to a product, not individual attributes.

How to do it...

1. Log in to **Magento Admin Panel** then go to **Catalog** | **Manage Products**.
2. Click on the **Add Product** button in the upper-right corner. You should be presented with a screen like the following:



3. For the simplicity of this guide we will stick to the **Default** attribute set and **Simple Product** product type. Please select both and click on the **Continue** button. The next screen will look as follows :



4. Enter all of the required fields then click on the **Save** button to save the product.

How it works...

If you try to click on the **Save** or **Save and Continue Edit** button before you have filled in all of the required data, JavaScript validation will trigger and circle all of the required input fields in red. So for example, if you filled all of the options under **General** and click on **Save**, JavaScript will take you to the next screen with missing required options.

Even when you enter all the required fields, chances are your product will not show up on the frontend store. There are several fields that are not marked as required but are needed, or some fields whose value is preset such that it indicates a non-visible product. If you are new to Magento this can be a bit frustrating to get right, since you simply do not see your product on the frontend even though you expect it to be visible.

Here is the list of fields that are truly required in order for your product to be visible on the frontend:

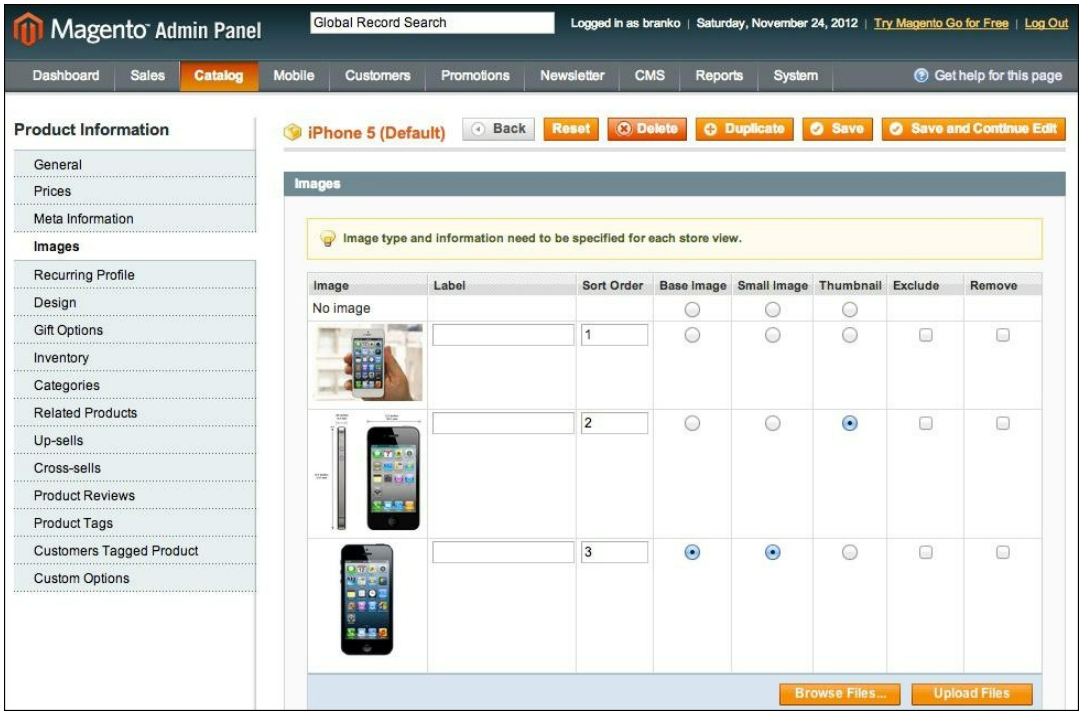
- In the **General** tab:
    - **Name**: Enter some text in this field
    - **Description**: Enter some text in this field
    - **Short Description**: Enter some text here
    - **SKU**: Enter some text in this field
    - **Weight**: Enter an integer or decimal value like 49.12345 (weight unit is dependent on the selected shipping method, meaning it's sometimes in pounds and sometimes in kilograms)
    - **Status**: Set it to **Enabled**
    - **Visibility**: If not sure what to use, use **Catalog, Search**
- In the **Prices** tab
    - **Price**: An integer value like 199.99
    - **Tax Class**: Select one from the dropdown (**None** and **Taxable Goods** are two common choices)
- In the **Inventory** tab
    - **Qty**: Some integer value
    - **Stock Availability**: Select **In Stock** from the dropdown
- From the **Categories** tab, in the **Product Categories** section select at least one category

Please note that in this example we are talking about clean Magento installation, but be prepared for the possibility that tabs might not always be 100 percent same, in that some options might be found under other tabs that are not exactly the ones listed here.

> ## Note
>
> In case you have multiple websites defined, the **Websites** tab will appear on the product edit page. In that case **Websites** also becomes the required option or else your product won't be visible on the frontend.
>
> Magento will automatically, based on the product name, assign the value for **Url Key** in the **General** tab if one is not provided.

There's more...

Besides just adding the required options to the product, you should also add some images to it. No one likes to see the default Magento **No image** image on the product view page on the frontend. Images are added from the **Images** tab as shown in following screenshot:



All images you try to upload should get written under the `/media/` folder. Thus if you experience image upload issues from within Magento you should check the folder permissions of the `/media/` folder; they should be 777.

You should now have a fully functional Magento product created and ready for sale.

Payment and shipping method (Must know)

Magento CE comes with several Payment and Shipping methods out of the box.

Since total payment is calculated based on the order and shipping cost, it makes sense to first define our shipping method.

The available shipping methods can be found in **Magento Admin Panel** under the **Shipping Methods** section in **System** | **Configuration** | **Sales**. **Flat Rate**, **Table Rates**, and **Free Shipping** methods fall under the category of static methods while others are dynamic. Dynamic means retrieval of rates from various shipping providers. Static means that shipping rates are based on a predefined set of rules. For the live production store you might be interested in obtaining the merchant account for one of the dynamic methods because they enable potentially more precise shipping cost calculation in regards to product weight.

Clean installation of Magento CE comes with the **Flat Rate** shipping method turned on, so be sure to turn it off in production if not required by setting the **Enabled** option in **System** | **Configuration** | **Sales** | **Shipping Methods** | **Flat Rate** to **No**. Setting up the dynamic methods is pretty easy; all you need to do is to obtain the access data from the shipping provider such as FedEx then configure that access data under the proper shipping method configuration area, for example, **System** | **Configuration** | **Sales** | **Shipping Methods** | **FedEx**.

Payment method configuration is available under **System** | **Configuration** | **Sales** | **Payment Methods**.

Similar to shipping methods, these are defined into two main groups, static and dynamic. Dynamic in this case means that an external payment gateway provider such as PayPal will actually charge the customer's credit card upon successful checkout. Static simply means that checkout will be completed but you as a merchant will have to make sure that the customer actually paid the order prior to shipping the products to him.

Clean installation of Magento CE comes with **Saved CC**, **Check / Money order**, and **Zero Subtotal Checkout** turned on, so be sure to turn these off in production if they are not required.

How to do it...

To configure the shipping method:

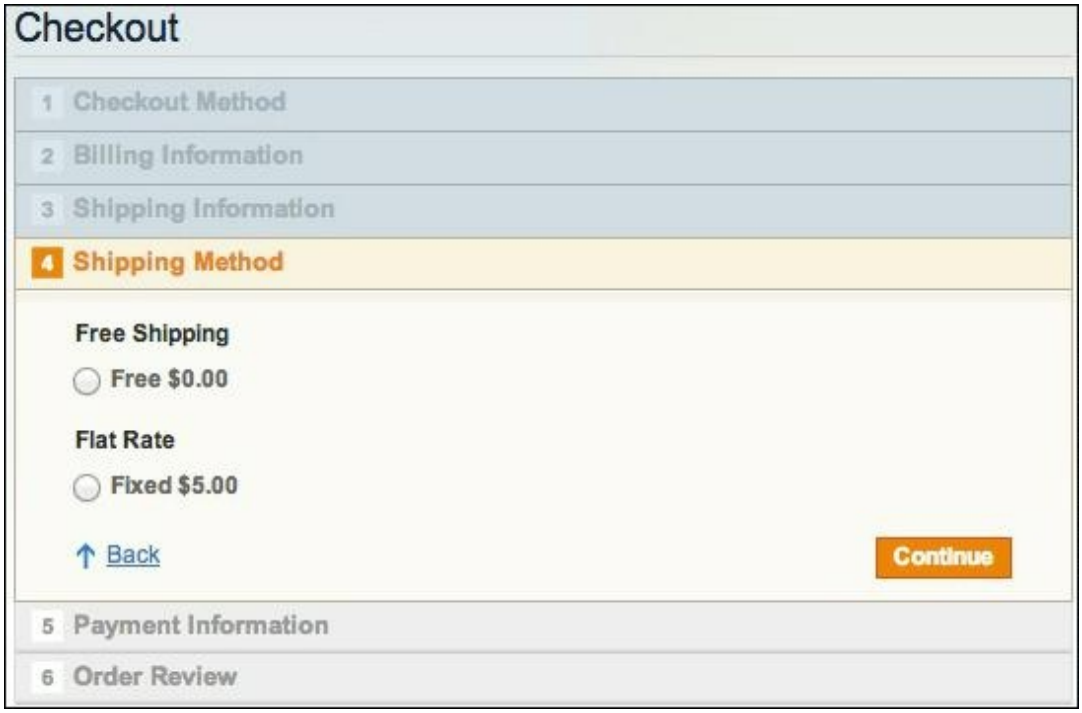
1. Log in to the **Magento Admin Panel** and go to **System** | **Configuration** | **Sales** | **Shipping Methods**.
2. Select an appropriate payment method, configure its options, and click on the **Save Config** button.

To configure payment method:

1. Log in to the **Magento Admin Panel** and go to **System** | **Configuration** | **Sales** | **Payment Methods**.
2. Select an appropriate payment method, configure its options, and click on the **Save Config** button.

How it works...

Once a certain shipping method is turned on, it will be visible on the frontend to the customer during the checkout's so-called **Shipping Method** step, as shown in the screenshot that follows. The shipping method's price is based on the customer's shipping address, products in cart, applied promo rules, and possibly other parameters.



Numerous other shipping modules are provided at Magento Connect on the site http://www.magentocommerce.com/magento-connect/integrations/shipping-fulfillment.html and new ones are uploaded often so this is by no means a final list of shipping methods.

Once a certain payment method is turned on, it will be visible on the frontend to the customer during the checkout's so-called **Payment Information** step, as shown in the following screenshot:



Additionally, there are numerous other payment modules provided at Magento Connect on the site at http://www.magentocommerce.com/magento-connect/integrations/payment-gateways.html.

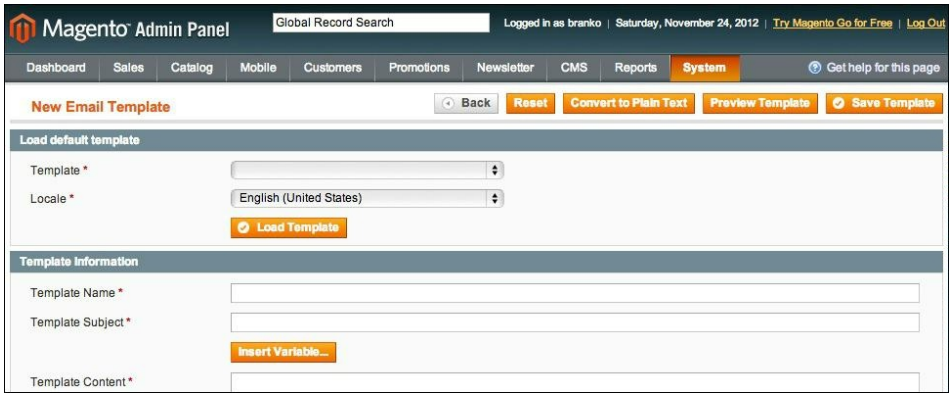
Transactional e-mails (Must know)

One of the most ignored features of Magento is its transactional e-mails. Transactional e-mails are all the e-mails sent from Magento. Unlike many other systems, Magento offers flexible, easy-to-use functionality to enable you to customize these e-mails and translate them into multiple languages to best use them for communication with your customers.

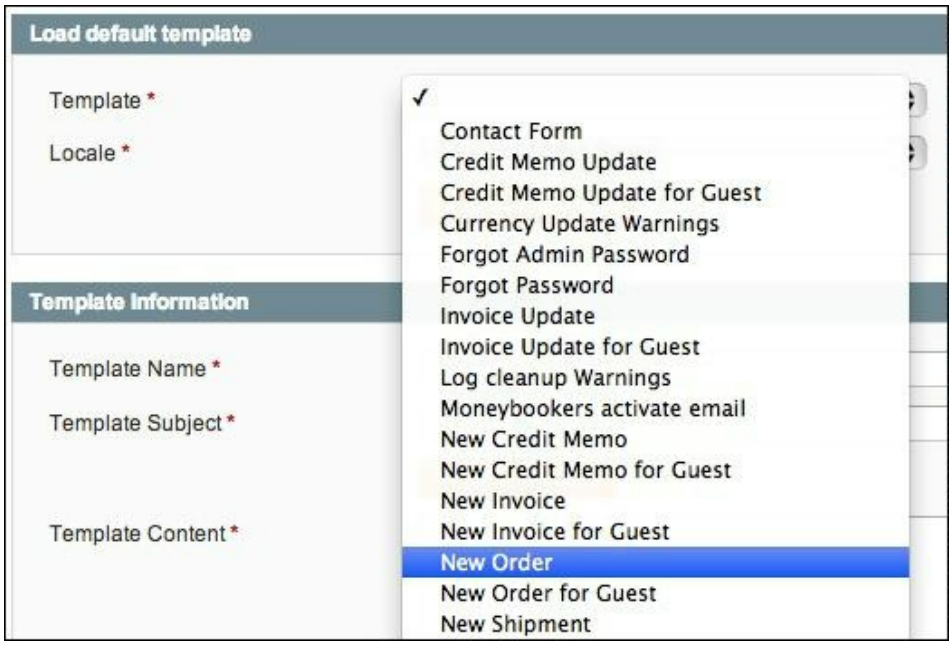
What does this mean? It means that you can edit the content of every transactional e-mail sent from Magento to suit your needs. This way you do not have to conform to the default e-mail content possibly hardcoded into the code of your system.

How to do it…

1. Log in to Magento backend administration, then go to **System** | **Transactional Emails**.
2. Now click on **Add New Template** in the upper-right corner of the screen. This should take you to the following screen:



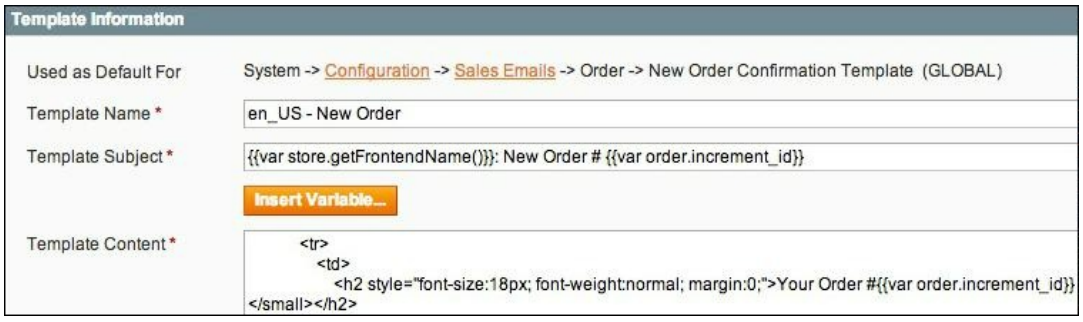
3. The next step is to choose the proper e-mail template from the **Template** option dropdown. The following screenshot shows some you can choose from:



4. Once you select the template you wish to edit, the next step is to select the **Locale**, then click on the **Load Template** button. This will trigger the loading of the transactional e-mail content into the **Template Subject** and **Template Content** fields.
5. Edit these fields to suite your needs and just click on the **Save Template** button when done.

How it works...

Template variable calls are done like `{{var order.increment_id}}`. Keep in mind that there is a limit to the variables available within individual templates. These variables sare injected into the template through the code and usually you have at your disposal only those variables that are found within the original e-mail template, as shown in the following screenshot:



Note

When naming your templates it is suggested to use the naming pattern similar to `locale - orig template name`. For example, if you edited

the **New Order** e-mail for **English (United States)** locale then name your template `en_US - New Order`. This approach will greatly increase the visibility of your templates under the **System** | **Transactional Emails** grid, especially if you have multiple locales installed.

Catalog and Shopping Cart Price Rules (Must know)

Catalog Price Rules are applied to the products before they are added to the cart while Shopping Cart Rules are applied to the products in the shopping cart.

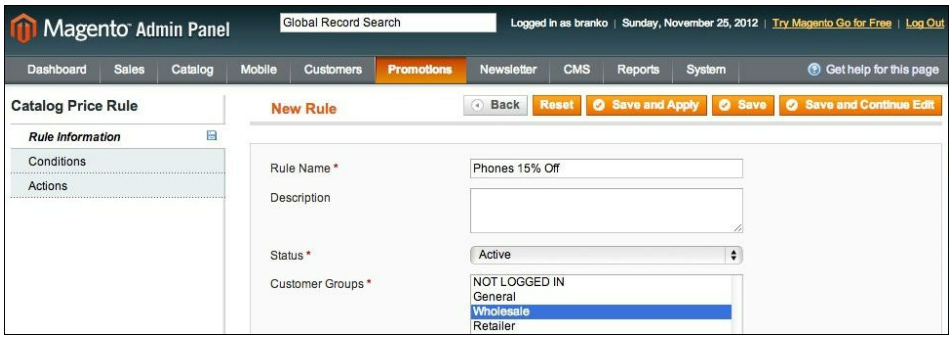
Getting ready

We will check out both Price Rule types by creating a useful case for each. We will use Catalog Price Rule to achieve *15 percent Phones discount for all wholesale customers*. We will use Shopping Cart Rule to achieve *Free shipping for orders over $100*.

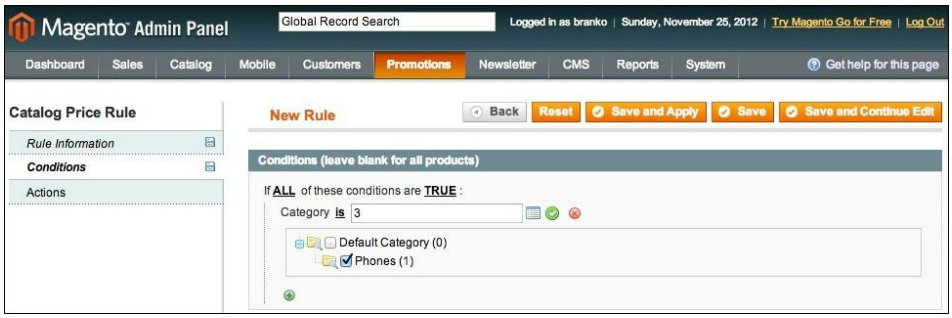
How to do it...

To create a *15 percent Phones discount for all wholesale customers* Catalog Price Rule follow these steps:

1. Log in to Magento backend administration, then go to **Promotions** | **Catalog Price Rules**. Now click on the **Add New Rule** button in the upper-right corner, you should be redirected to the following screen:



2. There are three tabs of options available. The information required here is pretty straightforward. On the **Rule Information** tab set `Phones 15% Off` for **Rule Name**, set **Active** for **Status**, select **Wholesale** for **Customer Groups,** then switch to the **Conditions** tab shown in the following screenshot:



3. On **Conditions** click on the little green plus sign then select **Category** under **Product Attribute** from the dropdown list; look for the **Phones** category and select it. Finally, switch to the **Actions** tab as shown in the following screenshot:



4. Under the **Actions** tab use **By Percentage of the Original Price** for **Apply** and set `15` for **Discount Amount**; leave the rest of the options on default and click on the **Save and Apply** button. You are done!

Customers who are logged in on the frontend and who belong to the Wholesale customer group will see 15 percent discounted product prices in the catalog. They will see the regular price struck through and the sale price listed next to it.

To create a *Free shipping for orders over $100* Shopping Cart Price Rule follow these steps:

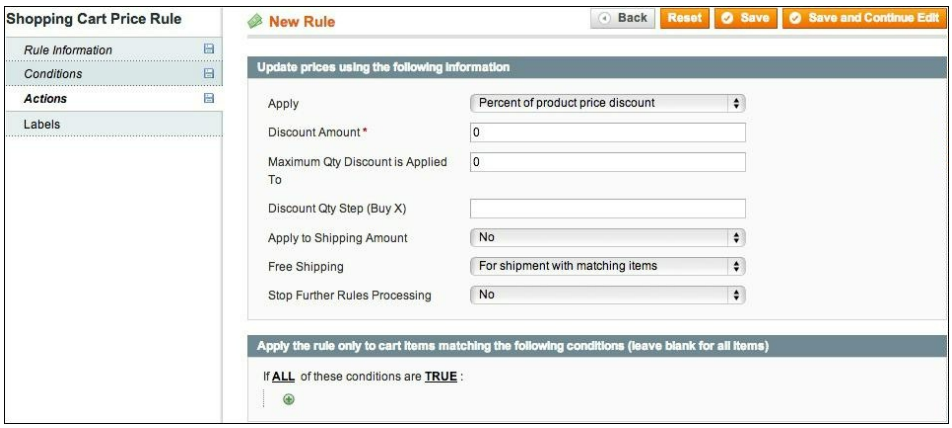
1. Log in to Magento backend administration, then go to **Promotions** | **Shopping Cart Price Rules**.
2. Click on the **Add New Rule** button in the upper-right corner, you should be redirected to the following screen :



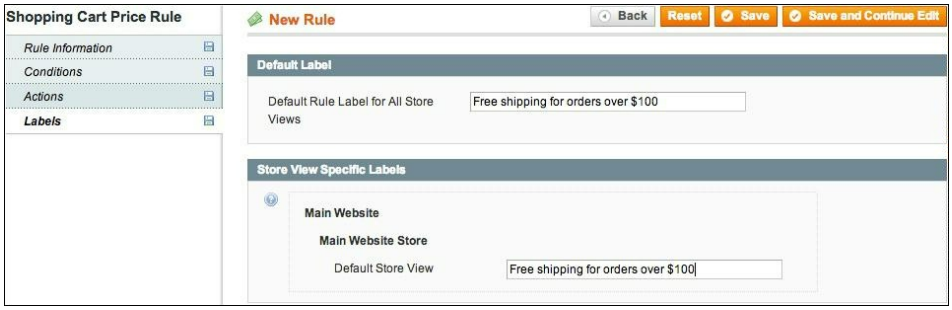
3. There are four tabs of options available. Information required here is mostly straightforward. On the **Rule Information** tab, set the **Rule Name** to `Free shipping for orders over $100`, **Status** to **Active**, **Customer Groups** to **General**, **Coupon** to **No Coupon**, then move to the **Conditions** tab as shown in following screenshot:



4. On the **Conditions** tab, click on the little green plus sign then select **Subtotal** under **Cart Attribute**, further select **equals or greater than**, then enter `100`. When done move to the **Actions** tab as shown in following screenshot:



5. Even though the **Actions** tab has several options we will only set the **Free Shipping** option to **For shipment with matching items** then move to the **Labels** tab as shown in the following screenshot:



6. On the **Labels** tab we basically need to set only the **Default Label** section; let's set it to `Free shipping for orders over $100`. We should now have all of the required options filled that conform to our desired `Free shipping for orders over $100` rule. Now click on the **Save** button.

Once the rule is saved, every customer who belongs to the customer group "General" will get free shipping if his order is equal to or higher than $100 (or some other base currency).

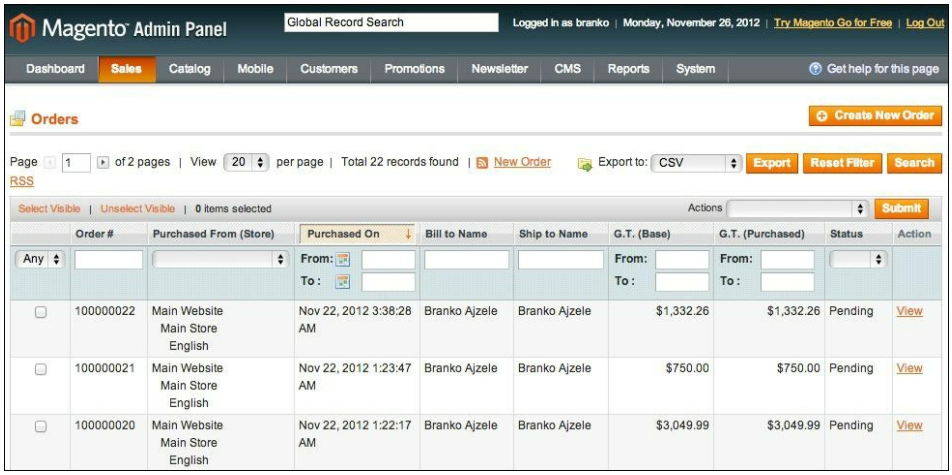
Managing orders (Must know)

Managing orders is probably the main task of every successfully-running web store.

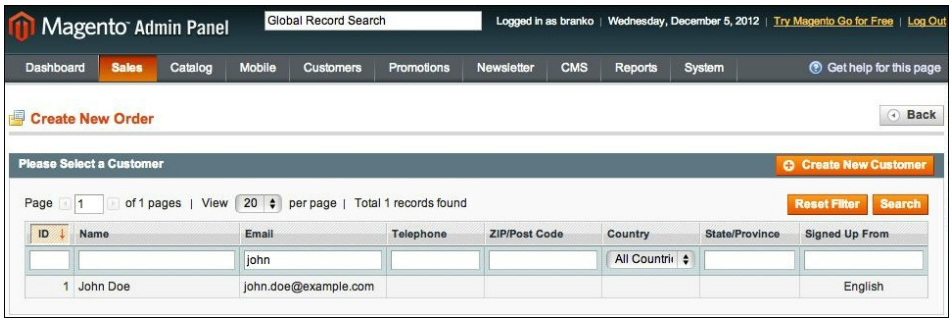
Magento gives you the ability to create an order in the administrative interface, just like your customer does in the frontend.

How to do it...

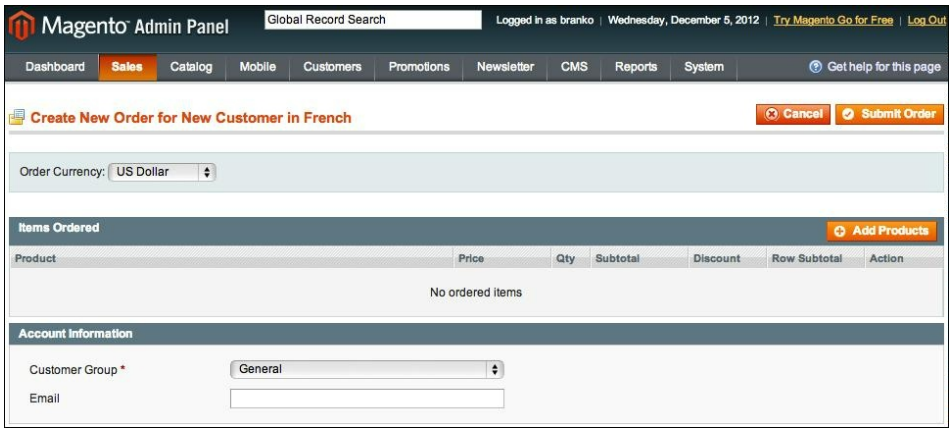
1. Log in to Magento backend administration and go to **Sales** | **Orders**, and you will see the following screen:



2. Click on the **Create New Order** button in the upper-right corner of the screen. This should initiate the **Create New Order** process.
3. You should now see a screen similar to that shown in the following screenshot. Now select a customer for which you are creating an order, or click on the **Create New Customer** button if the customer does not yet exist in the store. Both actions will instantly get you to the next screen.



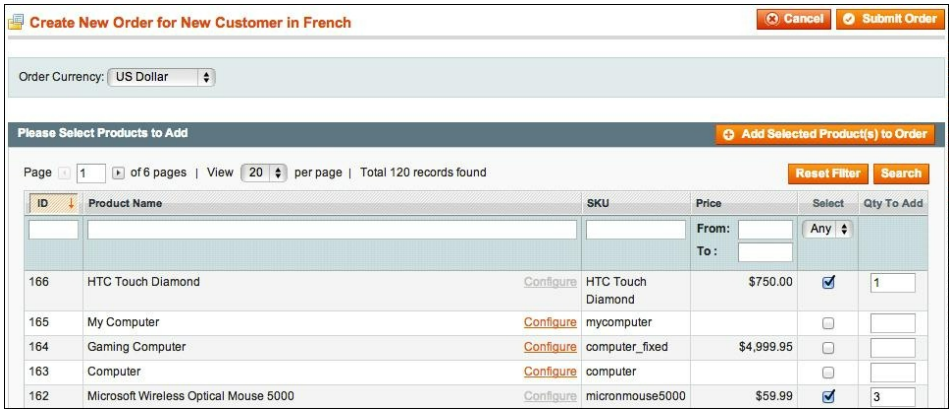
4. You should now see the following screen. This is the page where you provide the order details.



Note

The order page provides several sections for data entry, namely, **Items Ordered**, **Account Information**, **Billing Address**, **Shipping Address**, **Payment Method**, **Shipping Method**, **Order History**, and **Order Totals**.

5. The **Items Ordered** section is where we are adding products to the order. Just click the **Add Products** button and product selection grid should appear. Pay attention to the **Select** and **Qty To Add** columns in this grid. When you are done checking the products to add, click on the **Add Selected Product(s) to Order** button as shown in the following screenshot:



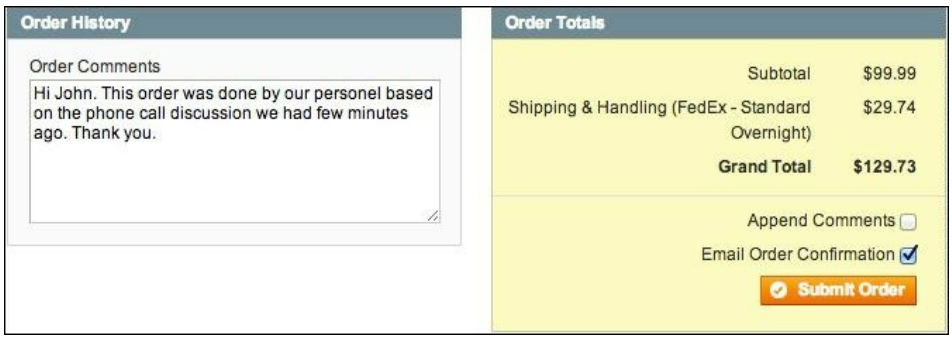
6. Move on to the **Billing Address** and **Shipping Address** sections as shown in the following screenshot. These are pretty self-explanatory. Be sure to enter both addresses correctly. You will notice how you switch between input fields during data entry. Magento will trigger the AJAX-ed loader and refresh certain parts of a page. This is normal, although it sometimes slows you down.



7. After you enter **Items Ordered**, **Account Information**, **Billing Address**, and **Shipping Address** you will be able to choose **Payment Method** and **Shipping Method** information as shown in the following screenshot. The **Shipping Method** section will likely need you to click on the **Get shipping methods and rates** link in order to load all of the available methods.



8. When you are done selecting the payment and shipping methods, it's always a good idea to throw in a comment under the **Order History** section as shown in the following screenshot (just be sure to check the **Append Comments** checkbox). Finally, check your **Order Totals** and if all is fine, click on the **Submit Order** button. This should create a new order for the customer. Be sure to check the **Append Comments** option, otherwise your order comment won't be visible to the customer.



How it works…

Take some time to get yourself familiar with the available grid columns so that later you know how to easily find individual orders. Usually the starting point for order management is the order `Status` value. If the order status is **Complete** this usually implies that all of the management around the order is already done (order paid, invoiced, shipped). There are several order statuses available.

Here is the list of the order statuses available right after installation:

- Processing [`processing`]
- Pending Payment [`pending_payment`]
- Payment Review [`payment_review`]
- Suspected Fraud [`payment_review`]
- Pending [`new`]
- On Hold [`holded`]
- Complete [`complete`]
- Closed [`closed`]
- Canceled [`canceled`]
- Pending PayPal [`pending_paypal`]

Next to each status, in the square brackets, is the list of accompanying states. Status is a child of state. You can think of a status like a nice label that is shown to the admin user and customer while a state is the code name used in the code itself. Each state can have several statuses (labels) assigned to it. You can define your own order statuses under the **System** | **Order Statuses** page.

As noted, order view page is divided into several sections; the following is a short explanation of each section.

The **Information** tab is the default one, as it contains the most important order information such as order number, customer account information, billing address, shipping address, payment information, shipping and handling information, items ordered, comments history, and order totals.

The **Invoice** tab holds the information about all invoices created for this order. An invoice is a record of the receipt of payment for an order. Invoices can be generated individually or together with shipments. You can have multiple invoices per single order. Each time you do an invoice you can simply invoice all or some of the items' full or partial sum of quantities within the order.

Invoicing an order means creating an invoice for all or just a subset of products found in the order. For example, if an order had two different products, one of them purchased in two quantities and another in one quantity, you could easily create an invoice for both in one quantity, which would leave one quantity of the first product non-invoiced. Later you can

come by and invoice the remaining quantity. Usually you would want to invoice only the orders with *Processing* state (in case of dynamic payments). If you invoice an order partially or fully its status will remain as **Processing**. Only when the order is fully invoiced and fully shipped, will the order status become **Complete**. To initiate the invoice of an order, simply click on the upper **Invoice** button in the upper-right corner of the screen.

The **Credit Memos** tab holds the information about the refunds given for items within the order. A credit memo is a record of a refund. Refunds can be given only on paid products. A product is considered paid once it has been invoiced. It is not a real refund in financial terms; you still need to give customers their money back. This is just a record of the refund for you.

The **Shipments** tab holds the information about all shipments created for this order. You can have multiple shipments created per order. Some shipping gateways, the dynamic ones, often come with a shipment-tracking functionality. A tracking number enables the customer to see the shipment status under **My Account**. Additionally, if supported by the shipping provider and Magento implementation, you can print shipping labels for each shipment.

The **Comments history** tab holds the information about order comments throughout the lifecycle of an order. Most of the time when you execute an action on an order, like invoicing or shipping, an entry of that action is created under comments history.

Editing an order in Magento might not be exactly what you expect it to be. Editing an order simply means canceling the current order and creating a new one with the same starting items in it.

Canceling an order means setting its status to **Canceled**.

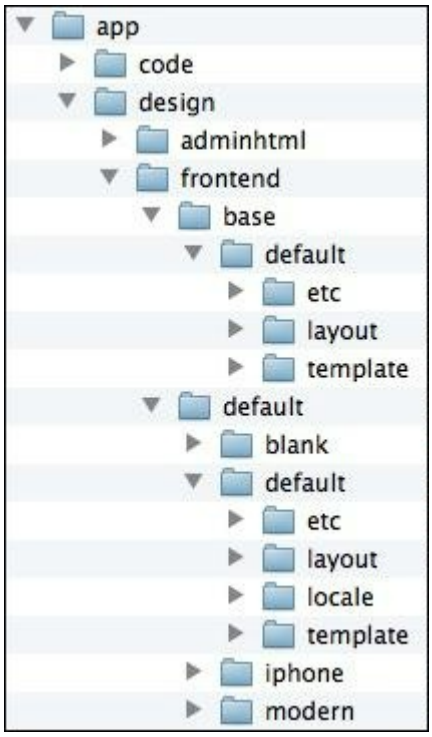
Holding an order means putting its status to **On Hold**.

Themes and appearance (Should know)

Magento supports pretty powerful and flexible theme functionality. In order to master it you first need to understand several key terms used in the theme discussions, terms such as block, static block, CMS page, skin file, template file, and layout file.

Getting ready

First, let's take a look at the structure of the Magento theme as shown in the following screenshot:

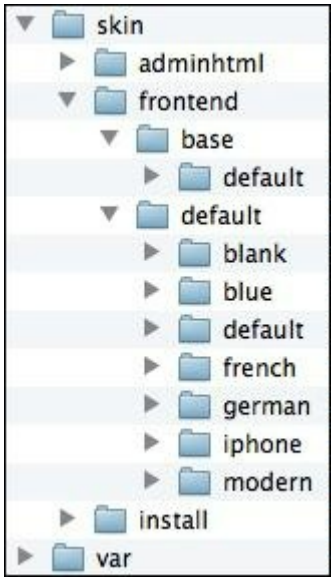


As you can see, themes can be found under the `/app/design/frontend/base` folder and under the `/app/design/frontend/default` folder.

In this case the `base` and `default` folders are called **package**. You can think of package as the topmost container for your themes. The way Magento works right after installation is that it will run the `/app/design/frontend/default/default` theme; let's reference it as `default/default`. However, the `default/default` theme does not actually contain all of the files required for the Magento frontend, so Magento will fallback to the `base/default` theme looking for missing files.

In other words, the `base/default` theme is the real default theme you get after the clean installation of Magento; it is also the theme to which Magento will ultimately fallback for files it cannot find in your custom theme. The `default/default` theme only contains minor modifications to `base/default`. You should never modify any file within the `base/default` theme.

Additionally, there is another root-level `skin` folder that is part of each theme structure as shown in following screenshot. You can see the naming pattern is the same.



The theme `etc` directory contains a `widget.xml` file, which is a loaded widgets XML configuration that is then merged with the theme widget configuration.

The `layout` theme's directory contains various XML files usually tied to single module functionality; for example, `checkout.xml`, `customer.xml`, `sales.xml`, and so on. Usually in custom made themes you will find the file named `local.xml` here.

The `locale` theme's directory contains subfolders like `en_US`, which further contain the translations in CSV files.

The `template` theme's directory contains various subdirectories of appropriate modules, which in turn contain various `.phtml` files. If you are building a custom theme this is place you will be spending most of the time, followed by the `layout` folder.

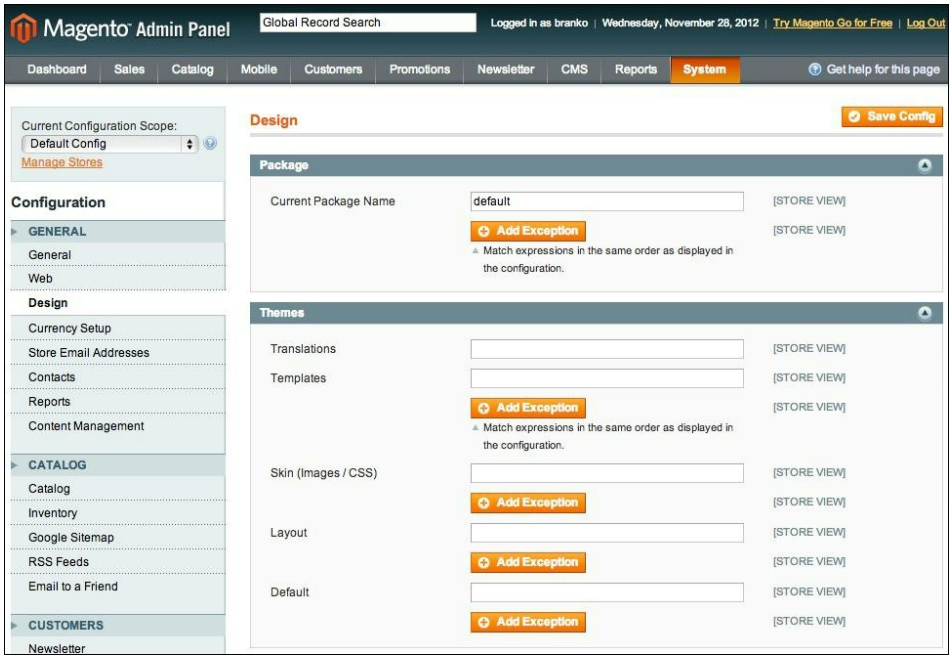
When you drill down into the `skin` folder, all the way to the individual theme, you will see a simple structure as shown in the following screenshot . These are pretty self-explanatory so we won't go into the details.



How to do it...

To add a new theme to your store and activate it, you need to do the following:

1. Download a new theme from the Magento Connect site at http://www.magentocommerce.com/magento-connect/themes.html.
2. Install the theme as you would any other Magento extension (using Magento backend administration, go to **System** | **Magento Connect** | **Magento Connect Manager**).
3. Activate the new theme by going to **System** | **Configuration** | **Design** and expanding the **Themes** section as shown in the following screenshot:



4. Specify the name of the new theme under the **Default** field.
5. Click on the **Save Config** button.

6. You might want to clean up the cache by going to **System** | **Cache Management**, selecting all cache types in the grid, and refreshing them via the **Refresh** action on the actions menu.