

VISUAL **QUICKSTART** GUIDE

Get up and running in no time!



Drupal **7**

TOM GELLER

© LEARN THE QUICK AND EASY WAY!

Visual QuickStart Guide

Drupal 7

Tom Geller

Peachpit Press

1249 Eighth Street

Berkeley, CA 94710

510/524-2178

510/524-2221 (fax)

Find us on the Web at: www.peachpit.com

To report errors, please send a note to: errata@peachpit.com

Peachpit Press is a division of Pearson Education.

Copyright © 2011 by Tom Geller

Project Editor: Nancy Peterson

Development Editor: Robyn G. Thomas

Copyeditors: Darren Meiss and Scout Festa

Technical Editor: Emma Jane Hogbin

Production Coordinator: Myrna Vladic

Compositor: David Van Ness

Indexer: Joy Dean Lee

Cover Design: Peachpit Press

Interior Design: Peachpit Press

Notice of Rights

All rights reserved. No part of this book may be reproduced or transmitted in any form by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the publisher. For information on getting permission for reprints and excerpts, contact permissions@peachpit.com.

Notice of Liability

The information in this book is distributed on an "As Is" basis, without warranty. While every precaution has been taken in the preparation of the book, neither the author nor Peachpit Press shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the instructions contained in this book or by the computer software and hardware products described in it.

Author photo courtesy Michel Chagall. All rights reserved.

Trademarks

Visual QuickStart Guide is a registered trademark of Peachpit Press, a division of Pearson Education.

Drupal is a registered trademark of Dries Buytaert. Macintosh and Mac OS X are registered trademarks of Apple, Inc. Microsoft, Windows, Windows XP, and Windows Vista are registered trademarks of Microsoft Corporation in the United States and/or other countries. UNIX is a registered trademark of The Open Group.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and Peachpit was aware of a trademark claim, the designations appear as requested by the owner of the trademark. All other product names and services identified throughout this book are used in editorial fashion only and for the benefit of such companies with no intention of infringement of the trademark. No such use, or the use of any trade name, is intended to convey endorsement or other affiliation with this book.

ISBN 13: 978-0-321-61921-1

ISBN 10: 0-321-61921-8

9 8 7 6 5 4 3 2 1

Printed and bound in the United States of America

Table of Contents

	Introduction	v
Chapter 1	Getting Drupal Up and Running	1
	Fulfilling Drupal's Requirements	2
	Downloading and Unpacking Drupal	8
	Creating the MySQL Database	
	Using phpMyAdmin	14
	Installing Drupal	17
Chapter 2	Establishing Your Drupal Site	21
	Using the New Administrative Interfaces	
	in Drupal 7	22
	Turning on Built-In Features	28
	Giving Your Site Its Identity	29
	Selecting a Visual Theme	33
	Monitoring Your Drupal Site	40
	Packaging Your Drupal Site	43
Chapter 3	Creating and Managing Content	51
	Gaining More Control of Individual Nodes	52
	Creating Other Types of Content	61
	Finding, Editing, and Deleting Content	71
Chapter 4	Customizing Content	73
	Defining Custom Types of Content	74
	Putting Images and Styled Text in Content	85
Chapter 5	Making Content Interactive	95
	Enabling Interactive Content Types	96
	Categorizing Content with Taxonomies	104
	Mastering Text Formats	110
	Mastering Image Styles	117

Chapter 6	Improving Access to Content	121
	Making Content Searchable	122
	Directing Traffic with Menus	128
	Laying Out Your Site with Blocks	140
Chapter 7	Wrangling Users	147
	Managing User Accounts	148
	Controlling How Users Interact with Their Accounts	156
	Defining User Roles and Permissions	163
	Building and Protecting Your User Community	171
Chapter 8	Customizing Drupal's Look and Feel	179
	Creating a New Theme	180
	Changing Theme Graphics and Typography with CSS	184
Chapter 9	Extending Drupal with Modules	193
	Using Modules	194
	Modules: The Drupal 7 Challenge	202
	Resources for Evaluating Modules	205
Appendix	Getting (and Giving) Help	211
Glossary	Drupal Terms and Culture	229
	Index	235

Introduction

With version 7, the web-publishing system Drupal stands at that frightening moment when it either enters the mainstream or falls into the abyss of niche enthusiasts. Until now its complexity has kept it mostly in the province of technology professionals, but I believe Drupal 7 will succeed for two reasons: The public is ready for it, and it's ready for the public.

First, the average web site builder understands—and demands—features that plain old HTML can't offer without extensive programming, such as user management and form processing. Drupal provides those features. Its architecture is elegant, its features broad and varied, and its community base unparalleled in the open-source software world (except by Linux, whose lead Drupal trails in terms of contributors).

But it's not the only contender, and in fact several others are easier to use. Drupal's boosters (*Drupalistas*) point out that their software does well against Joomla and other full-featured CMSes (content management systems), but I think they're missing the point: The real competition is

anything that gives people the features they want. So for simple personal sites, it's Facebook and MySpace; for businesses, Yahoo Merchant Solutions, eBay, and Amazon; for publications, WordPress.com and Blogger.com; and for the rest, a hosted solution like Google Sites. Drupalistas would argue that none of these are technically CMSes. It doesn't matter—they have enough CMS-like features for the masses.

But time spent learning Drupal rewards its student handsomely. None of those other solutions compares to it in terms of flexibility, security, or support. Learning to exploit Drupal's advantages just takes time and attention.

I wrote this book because I wanted to help non-technical people discover the joy I found when I started building sites in Drupal in late 2007. Freshly downloaded, Drupal gave me the ability to run a blog (like WordPress) and host polls (like MySpace) in addition to everything I could do before with plain HTML. Dipping into the vast wealth of Drupal extensions (*modules*) gave me plug-and-play access to features

I hadn't even considered before. My first Drupal site (savemyhomebook.com) looked and functioned far better than anything I'd built since learning HTML in the mid-'90s; later sites included shopping carts, user interaction, and complex data display.

Which brings us to the second reason Drupal 7 is right for non-technical site builders: Its developers have focused on making it the easiest version of Drupal. In early 2009, project founder *Dries Buytaert* made it clear that he wanted "radical improvements in usability" to this version, even funding design improvements through his company, Acquia. Eye-tracking studies at the University of Baltimore pinpointed items of greatest confusion among first-time Drupal administrators, and outreach programs encouraged the participation of graphic designers and user-interface experts.

The results have fallen short of some of the more ambitious goals, but are impressive nonetheless. The Drupal community (as the saying goes) shot for the sky and missed—but landed among the stars.

Is This Book for You?

Drupal 7: Visual QuickStart Guide was written for anybody who wants to create a dynamic, easy-to-update web site that looks good and performs well. Unlike most current books about Drupal, it's written for anybody with even the most basic computer skills. That means you should be already able to:

- Use a computer to access the Internet through a web browser
- Download files
- Install software on your computer

Depending on how you intend to host your Drupal web site, you might also need to:

- Access a remote computer through programs other than a web browser
- Navigate by typing on a non-graphical, command-line interface such as is found in *nix operating systems
- Change file permission settings
- Understand written instructions from your web hosting provider

What This Book Will Teach You

By the time you've finished this book, you'll have all the skills you need to create an attractive and complete Drupal web site. Specifically, you'll learn how to:

- Install Drupal on your home computer (to prototype your site)
- Install Drupal on a remote server (to make your site available to the world at large)
- Give your site its basic identity
- Add, change, and delete text and images
- Modify your site's visual design
- Modify your site's functional interface, for example changing where information appears on the page and what links show up in menus
- Monitor and maintain your site to prevent malicious activity
- Make a backup for safety, and restore your site from that backup
- Allow site visitors to become members with their own user names and passwords
- Control member access
- Run interactive features, such as blogs, polls, and forums

- Find and add any of hundreds of modules that extend Drupal with new and interesting functions
- Find help from experienced Drupal administrators when you get stuck
- Further your Drupal career by understanding and participating in the project's vibrant user community

What This Book Won't Teach You

Alas, there are some Drupal-related subjects too big or tangential to include here, such as:

- Finding and signing up with a web host to make your site available to the world. (There is a list of companies that provide Drupal-ready hosting services at drupal.org/hosting.)
- Accessing files on, transferring files to, or navigating around your remote host's file system. In Chapter 1, "Getting Drupal Up and Running," I briefly show how to navigate one of the most common host interfaces—the command-line interface of *nix. But there's an enormous variation in how web hosts operate and what they allow their customers to do. If the instructions in this book don't fit, ask the support desk of your web host for help.
- Registering a domain name or making it refer to your Drupal site. Again, your web host's help desk is the best place to go.
- Suggesting anything about the *kind* of content to put on your site. That's your job!
- Programming or advanced theming.

How This Book Is Organized

Drupal 7: Visual QuickStart Guide is intended to take you from a standing start to running a web site that's both functional and useful. Because the definition of "useful" varies from person to person, this book makes only basic assumptions about what you're going to do with Drupal. The book is divided into four sections based on those assumptions:

- I Setting Up (Chapters 1 and 2)
- II Managing Content (Chapters 3–6)
- III Managing Drupal (Chapters 7–9)
- IV Appendix and Glossary

As you can see, the biggest section is devoted to creating and managing content. That's probably why you're learning a *content management* system! But this book also assumes that you'll want to:

- Open your site to a wider community, and therefore need to understand Drupal's user-management features (Chapter 7)
- Change your site's appearance (Chapter 8)
- Take advantage of Drupal's huge library of modules to extend what it can do (Chapter 9)
- Go beyond this book to interact with Drupal's user community, one of the most active in the open-source software world (Appendix)

In making these assumptions—and to best use limited space—I've left out some of Drupal's finer points. Typically these were features that will matter to you only after you've been administering Drupal sites for a while. For example, I decided not to

discuss performance or security enhancements, because Drupal's built-in features are sufficient for all but the busiest web sites. The appendix, "Getting (and Giving) Help," gives you pointers about where to find such information should you need it.

At times, I use a very simple, fictional Drupal site to demonstrate various points. If you'd like inspiration from people who have taken Drupal much further, see drupalsites.net.

Standards Used In This Book

I use some conventions to provide guidance. This book:

- Uses **this font** whenever showing something that you should actually type (also known as *code*).
- Shows the code font in italics to indicate that you should replace the text with the equivalent for your situation, for example, `http://domain-name/user/user-id`.
- Italicizes the first occurrence of words that are defined in the glossary.
- Refers to locations on your own Drupal site in the form `http://domain-name`.
- Refers to other web sites in the form `example.com` (without the `http://`).
- Refers to directory paths on your web server in the form `/path/to/location` (with a leading slash, which indicates the top directory of your Drupal installation).
- Assumes that you're always logged in as the *superuser* (`user/1`)—that is, the user that you created when you first installed Drupal. (For details, see Chapter 1.) This user has full access to every administrative feature in a Drupal site. If you get the message "Access

Denied" when you try to do something in this book, try logging out (by going to `http://domain-name/logout`) and then logging in again (`http://domain-name/user`).

- Gives instructions for Mac first, followed by Windows and *nix in those instances where differences among them arise. (Since you manage Drupal mostly through web browsers that work pretty much the same on all platforms, these distinctions are rare.)
- Provides screen shots of the **drupal.org** web site as it appeared immediately after its October 2010 renovation. In an example of bad timing, the site was undergoing profound reorganization as we were putting this book to bed. Some screen shots you see here are of the prerelease beta site, and might be different from what you see when you visit **drupal.org**.

What Is Drupal, Anyway?

Drupal is in the content management system (CMS) category of web site tools. CMSes excel in letting you create *dynamic* sites, which show different information depending on a number of factors, such as the input of previous visitors, or whether the current visitor is logged in. Sites that don't have such features are called *static* and are typically created using a page-description language such as HTML. (You can also use Drupal to create sites that don't take advantage of its dynamic features.)

Drupal has many features common to CMSes, including:

- Administration through a web browser. You manage your Drupal site mostly

by visiting it in a web browser such as Firefox or Safari, logging in as an administrator, and going to pages that let you change site settings.

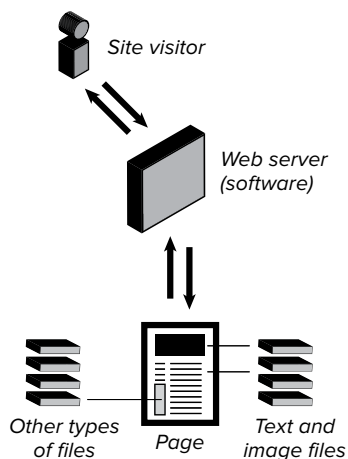
- A user-management system that lets you identify, track, and control visitors' access.
- Fine-grained permissions that allow you to grant specific rights to specific groups of users.
- Streamlined methods for changing content. In Drupal, you edit basic pages (or other types of content) by clicking a tab labeled Edit and filling out a form. To do the same thing on a traditional HTML site, you'd need to download a file, figure out what parts to change, make the changes, and then upload the file again.
- Flexible methods of displaying content. One example is Drupal's Summary feature, which shows a shortened version of content where appropriate.
- Consistent appearance throughout the site. Certain features (such as menus and graphic design) remain the same regardless of what part of the site you visit.
- Changeable overall appearance. In Drupal, you change the look of the site as a whole by switching to a new *theme*, hundreds of which are available from **drupal.org** or through private designers. The content remains the same regardless of what theme you select.
- Extensibility so you can add features by writing (or downloading) a bit of programming code, typically in the PHP language. Drupal is usually extended through the use of modules, which you can read about in Chapter 9, "Extending Drupal with Modules."

How Drupal Works

The best way to understand how Drupal works is to compare it with other systems, specifically static HTML and hosted sites.

- Static HTML sites comprise text files that end in **.htm** or **.html**; image files (**.jpg**, **.gif**, **.png**); other media and downloadable files such as Adobe Portable Document Format files (**.pdf**) or QuickTime movies (**.mov**); and any custom programming files. These reside on a remote computer (the *host*) that's always on and connected to the Internet and that runs a program called a web server.

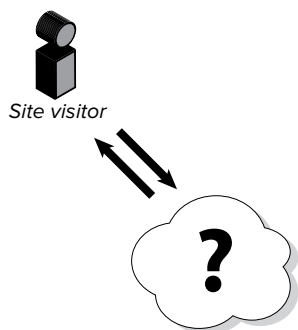
When someone types the domain name of a static HTML site into a web browser, the web server finds a file called **index.html** (or **index.htm**) on the host. That file typically contains all the text on the site's home page, along with references to other files such as images and formatting information. The web server gathers all these pieces together and sends them back over the Internet to the web browser, which reassembles them into a web page **A**.



A How static HTML sites produce pages

To change a static HTML site, you either modify files directly on the server or on your personal computer and then transfer them to the server through a program such as FileZilla (filezilla-project.org), Secure Copy (SCP), or a web browser–based interface. Likewise, to copy your HTML site to another computer, you only need to copy its files.

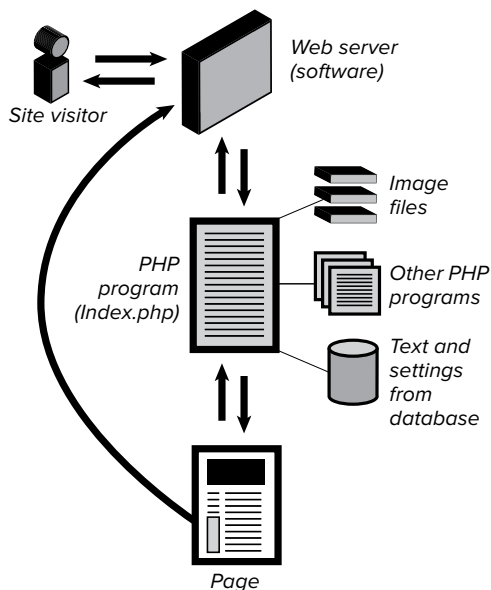
- Hosted sites such as Yahoo Merchant Solutions and Google Sites vary widely in how they’re set up, but they share some common traits. For one thing, you almost always build and modify your site through a simplified, web browser–based interface that hides the underlying HTML and programming code from you. As a result, such solutions tend to be easy to create but hard or impossible to customize beyond pre-defined limits. You neither need to know nor are able to know exactly *how* the site runs beyond what you can see through the web browser **B**. It’s sometimes impossible to copy a hosted site to another computer in a useful form—you’re stuck with the host for life.



B How hosted sites produce pages (as far as we know)

- Drupal sites essentially comprise three parts: the Drupal software itself, which is written mostly in the PHP programming language; a database that contains your site’s content and settings; and images and other files.

When someone visits a Drupal site, the web server first finds the **index.php** file. Whereas a static HTML site’s **index.html** file simply describes what the page should look like, Drupal’s **index.php** file is a program that causes the host to look into dozens of files and the content database to determine what information should be sent to the visitor. When the process is finished, the web server responds with files from the host and content from the database, formatted (mostly) as simple HTML. The web server sends that HTML over the Internet to the web browser, which interprets and displays it just as if it had come from a static HTML site **C**.



C How Drupal produces pages

Although this process seems impossibly complex to the human eye, the host interprets this chain of PHP and database instructions quickly and easily. Along the way, Drupal has opportunities to modify the output based on its own programming, modules you've installed, and any other circumstances.

You make most changes to a Drupal site by manipulating content in the database through a web browser-based interface. Except for Chapter 1, this book is almost entirely about how to use that web browser-based interface.

To copy a Drupal site to another location, you need to transfer both its files and its database. You'll learn how to do that in Chapter 2, "Establishing Your Drupal Site."

Drupal, CSS, PHP, JavaScript, and SQL

I mentioned that Drupal is mostly written in PHP, which was designed specifically for web development. Technically, PHP is a *scripting* language, which (among other things) means that programs written in it are stored in a form that you can easily read; an *interpreter* turns them into machine language at the time they're run. (By contrast, programs written in *compiled* languages such as C++ are converted into machine language before they're run, and then the computer uses the converted version from then on.) As a result, anyone who knows PHP can look at the Drupal program and understand how it works.

PHP is a *server-side* language, meaning that all of this interpretation happens on the host—typically, on the same remote computer that stores the files and sends the final HTML code out over the Internet.

But PHP isn't the only language involved in a Drupal site. Others are:

- **SQL** (Structured Query Language), a server-side database language that Drupal uses to add, change, and remove information. When someone posts a page to your site, for example, Drupal sends a command in SQL to the database that says, "insert this information into the **node** table" (actually, Drupal sends several SQL commands). When somebody reads that page, Drupal says, "retrieve that information from the **node** table." Here's an example.

```
INSERT INTO `node` (`nid`, `vid`,  
→`type`, `language`, `title`, `uid`,  
→`status`, `created`, `changed`,  
→`comment`, `promote`, `moderate`,  
→`sticky`, `tnid`, `translate`)  
→VALUES (1, 1, 'page', '', 'Title  
→goes here.', 1, 1, 1257477418,  
→1257477418, 0, 0, 0, 0, 0, 0);
```

- **CSS** (Cascading Style Sheets), a descriptive language that defines typography, layout, and other display properties. One of the fastest ways to give your Drupal site the appearance you want is to download a free theme that has the basic layout you like, and then change its CSS files to indicate your preferred fonts, colors, and images. (You'll learn how to do this in Chapter 8, "Customizing Drupal's Look and Feel.")

Unlike SQL and PHP, CSS is a client-side language that's sent over the Internet and then interpreted by site visitors' web browsers. Here's an example.

```
h1, h2, h3, h4, h5, h6 {  
    margin: 0;  
    padding: 0;  
    font-weight: normal;  
    font-family: Helvetica, Arial,  
    sans-serif;  
}
```

- JavaScript, a language that mostly adds interface “spice” to your site. If an image changes when you move your mouse over it, that’s JavaScript at work. Drupal itself includes several user-interface portions of the jQuery library, a JavaScript extension that allows developers to easily include lots of “eye candy,” such as animations, resizable dialog boxes, and drag-and-drop effects.

Like CSS, JavaScript interpretation happens in site visitors’ web browsers, not on the server. A very small number of site visitors turn off JavaScript on their browsers and won’t be able to see these effects. Fortunately, Drupal always tries to present them the same information without the effects. Here’s some sample JavaScript.

```
Drupal.progressBar.prototype.  
→ startMonitoring = function  
→ (uri, delay) {  
    this.delay = delay;  
    this.uri = uri;  
    this.sendPing();  
};
```

You don’t need to learn any of these languages to use Drupal! But doing so lets you create unique features for your sites and improve your overall computer knowledge. Besides, Drupal provides both motivation and opportunity to learn them: Why not take advantage of it? I personally had a hard time wrapping my head around CSS—until I started modifying Drupal themes. Likewise, my real first forays into PHP were to display information that couldn’t be extracted in any other way.

Having said that, contributed modules sometimes take the place of these languages. For example, the Views module (which you can download for free from drupal.org/project/views) makes most custom SQL programming unnecessary.

The Long Road to Drupal 7

Drupal has come a long, long way since Belgian student Dries Buytaert started creating it in 2000 to stay in touch with his friends at the University of Antwerp. It received a big publicity boost in 2003 when supporters of U.S. presidential candidate Howard Dean organized campaign activities using Drupal 4, and since then it has grown by leaps and bounds.

By the time version 5 arrived in early 2007, Drupal ran such notable web sites as theonion.com and mtv.co.uk. Drupal 6 came a little over a year later, including features that eased installation, maintenance, content translation, and general administration.

Mr. Buytaert listed 11 key improvements he wanted when Drupal 7 development started in earnest in February 2008 (see buytaert.net/starting-to-work-on-drupal-7). As I mentioned, the one that got the most attention—and that’s most visible to anyone who’s tried earlier versions—is *usability*. That focus has frankly required a change within the community’s social structure, as the text-minded developers who make up its inner circle needed to see the value of good visual design and user interaction. But with Drupal 7, they’re beginning to see the light. They belatedly follow in the footsteps of Mr. Buytaert’s own conversion, which he described in a January 2006 post:

“For long I focused, completely and utterly, on the aesthetics of Drupal’s code, neglecting eye candy and ease of use... The aesthetics of Drupal’s clean code has attracted many developers, but has also given Drupal the reputation of being developer-centric and hard to use... I have since learned that elegant design and ease of use are equally [as] important as clean source code.”

A secondary goal of Drupal 7 was, put bluntly, to not repeat certain mistakes of the Drupal 6 launch. Three practices have given the Drupal 7 release plan a level of professionalism that any organization would envy:

- Drupal development tools are better. Testing, communication, and project tracking have all benefited from advances of the past two years.
- Drupal 7’s release schedule was defined better. A series of “code freezes” set deadlines that both revved motivation and imposed discipline. While those finish lines moved a few times (introducing the giggle-worthy phrase “code slush”), and there were occasions of blatant disregard for some of those deadlines, there’s no denying the clarity they attempted to give Drupal 7’s development process.
- Support by Drupal module developers is better. One of the biggest complaints surrounding Drupal 6’s release was that most modules—including those that nearly every Drupal site administrator needed—lagged the release by months or even years. This time around, Moshe Weitzman lobbied for developers to pledge release of their

Drupal 7-compatible modules for the same time that Drupal 7 itself is released. That effort has garnered over 100 pledges. Of the 20 most popular modules, only six have neither taken the pledge nor been incorporated into Drupal 7 itself. Even without the pledge, Drupal 7 versions of every one of those six are in active development. (For details, see “Modules: The Drupal 7 Challenge” in Chapter 9.)

Drupal 6 vs. Drupal 7

So what finally came out of the Drupal 7 sausage grinder? Dozens of changes, described in minuscule detail in the periodic release notes linked from the Drupal project page at drupal.org/project/drupal. Although you could rightly argue about which are the most important, here’s my take (summarized from my article on the Peachpit Press web site at peachpit.com/articles/article.aspx?p=1433049):

- New themes. Drupal 7 adds a new default theme (Bartik) as well as a separate theme, appropriately called Seven, that makes administration easier.
- Lighter workflow. For version 7, links to Drupal’s administrative functions have been radically reorganized into what its developers hope is a more user-friendly configuration, although it might leave Drupal 6 administrators scratching their heads for a while. Drupal 7 also puts commonly used commands in easy reach, in the Toolbar and customizable Shortcut bar. Finally—and most visibly—there’s now an administrative overlay that floats controls in front of the screen they affect, reducing the feeling that you’re navigating through a maze of screens.

- Improved installation and update procedures. What probably most often tripped up users of Drupal 6 was installation and updating, both of which have been fundamentally improved in Drupal 7. First, the installer has some nice touches that make it more fool-proof. Second, help texts during installation are a lot more, well, helpful. Once you have Drupal installed, you can now install and update themes and modules through Drupal's web-based interface instead of needing access to the server (via its sometimes obscure commands).
- Smarter defaults. User studies and administrator experience have made Drupal more ready for use immediately upon installation. You can still override such behaviors, but on the whole, Drupal administrators will do a lot less initial work to make their Drupal 7 sites work the way they want.
- Easier, more flexible content management. For a content management system, the procedure for entering content in Drupal has historically been more difficult than it needed to be. But Drupal 7 makes up for lost time with three standout features. First, a neat visual trick called “vertical tabs” makes the content-entry form cleaner and easier to navigate. Second, you no longer need to download additional modules to put images in content. Finally, you can define highly customized types of content, for example catalog pages that include *fields* for price, color, and size. Before Drupal 7, you needed a module called Content Construction Kit (CCK) for this functionality. (See Chapter 4, “Customizing Content,” for details.)
- Fields in core. Drupal 7 has many improvements that are hidden from view—unless you're an experienced administrator or Drupal developer. “Fields in core” is one of those changes, but it has far-reaching benefits for everyone who uses Drupal. It essentially lets you break information into separate parts (*fields*) for truly original results. Fields in core actually goes beyond content, extending *profiles*, comments, and how you categorize content.
- Easier programming. As long as I'm talking about less-visible improvements, I should mention a huge raft of changes that have made programming for Drupal easier. The list is far too long to include here, but you can see it at <http://drupal.org/node/224333> and <http://drupal.org/node/394070>.
- Clean-up of unneeded bits. Some things removed from Drupal 7 include the little-used Throttle, Ping, and Blog API modules; the “related terms” feature for taxonomy (which never did anything, anyway); and the ability to block posts that don't have a minimum number of words. Of course, every bit of obsolete technology has its fans, so such removals are always controversial. If you find yourself needing those old Drupal 6 features, most (if not all) will still be available through downloadable modules.
- Better organization of user permissions. Drupal has long offered extremely fine-grained permission controls. For example, you could allow users to edit (but not delete) their own blog entries. With that flexibility comes confusion, though, and the dozens of check boxes on Drupal 6's default permissions screen often threw new administrators for a loop. Those permissions have

been reworked for clarity, and there's a new "administrator" role.

- Contextual edit links. When you're logged in as the administrator, a Drupal 7 site now presents options as you hover your pointer over certain items you can control. For example: To edit a block in Drupal 6, you had to go to the Blocks configuration page, guess the name of the block you wanted to change, and click "configure"; in Drupal 7, you just point at the block and click. Much better!

Acknowledgments

"No man is an island," wrote Renaissance poet John Donne, whose separate Elegy XIX ("To His Mistress Going to Bed") is a gleeful celebration of erotic anticipation. So mix the profound and the profane, as they do in everyday life. These acknowledgments, too, mix thanks to both central and incidental influences. Some made the book what it is; some made me what I am. Which is which is left as an exercise for the reader.

First, the obvious. Robyn Thomas has been an attentive and friendly development editor: She kept me (comparatively) honest. Project editor Nancy Peterson gave difficult decisions about timing and content the attention they deserved, ensuring this book's relevance and longevity. Technical editor Emma Jane Hogbin (emmajane.net) has been the *real* Drupal expert here, insisting on (among other things) the correct spelling, usage, and politics of "*nix" until I got it right. [Emma: I *did* finally get it right, didn't I?] Greg Knadison (growingventuresolutions.com) took time away from his busy schedule to

step in when we needed some emergency fact-checking.

Darren Meiss and Scout Festa polished the copy swiftly and silently, like literary Rumpelstiltskins. When it was all finished, product marketing manager Glenn Bisignani ensured that the resulting volume would find its way into your hot little hands.

Thanks also go to the two people who got this project off the ground: my agent, Neil Salkind of the Salkind Literary Agency (salkindagency.com), and Peachpit's senior acquisitions editor Wendy Sharp. By the same token, I'd like to thank the mob of computer book professionals who have guided me to this point over the years, particularly those met through various technical communities and the Studio B discussion list.

This list doesn't include the many, many people involved in the book's production and distribution, some of whom toil with names unknown to me. That they're not explicitly included shouldn't suggest a diminishment of their services or of my gratitude.

Thanks for other reasons go to Joanne Brodie, Laura Sherwood, Lisa Carlotta, and Ali Leal, along with others who recognize a pattern and believe they should be in that list. It's cliché to say "You keep me sane," so here's a greater truth: You keep me happy.

It's shocking that this is my third book, yet I haven't thanked the *ultimate* originators of this project: my parents, Mimi and Conrad Geller. What kind of son would do such a thing? And comb your hair, it looks like a rat's nest.

I'd like to also thank the good people of Oberlin, Ohio, where I decided to make my home a few weeks before starting this

book. It was written almost entirely at the facilities of Oberlin College and the Oberlin Public Library, with occasional stints in several of the town's restaurants, bars, cafes, and lobbies, and in the Wi-Fi-enabled town square. I couldn't have chosen a better place.

Lastly, one more shot at the obvious:

This book would have no purpose, and the web would be a poorer place, if not for the tens of thousands of people who program, test, use, document, critique, and otherwise enjoy Drupal. Its success has put high-quality, 21st-century web publishing within the reach of millions of people who'd otherwise be stuck with harder and less-capable options. It's my sincere hope that this book continues the community's work in some small way and strengthens the project's goals.

4

Customizing Content

Drupal suffers from the “blind men describing an elephant” syndrome. In that Indian fable, one man touches the trunk and declares that an elephant is like a tree branch; another touches the tail and says an elephant is like a rope; a third touches an ear and says the elephant is like a great leaf; and so forth. The thing observed is seen by its parts, according to the needs of the individual.

Since Drupal’s audience has historically been software developers, that’s translated into strong support for text and add-on programming. Two areas where it’s not been as capable are in creating content types with custom fields, and in supporting graphics and text styling within nodes.

Fortunately, both areas are getting better. This chapter tells you how to create more flexible content using tools that are available today. However, a constant stream of new solutions passes through the library of contributed modules at drupal.org/project/modules. For details on how to find and use them, see Chapter 9, “Extending Drupal with Modules.”

In This Chapter

Defining Custom Types of Content	74
Putting Images and Styled Text in Content	85

Defining Custom Types of Content

Now we come to one of Drupal's most useful features: custom *content types*. Through them you can move far beyond built-in types and into a world bound only by your imagination. For example, an “employee” content type could include fields for position and salary, and a “catalog page” content type would have fields for prices, colors, and sizes.

Nearly all Drupal sites today have custom content types. But amazingly, Drupal succeeded for years without this feature. It was first available in 2005 through a contributed (non-core) module called Flexinode, which was replaced by Content Construction Kit (CCK) in 2006. Now, Drupal 7 incorporates most of CCK in its core.

Although this section teaches how to create new content types, the instructions are essentially the same if you want to edit an existing content type, such as an article or basic page. If you do so, be aware that changes to an existing content type will primarily affect only *newly created nodes* of that content type. If you add an age field to an existing content type, for example, and require that the age field contain a value, then existing nodes without a value in that field will be unaffected—at least until you go to edit the node. At that point, you'd be prompted to fill in the required field.

A The form for editing a content type

B Naming and describing a new content type, with the pointer indicating how to change its machine name

To create a new content type:

1. Click Structure in the Toolbar and then click “Content types.”
2. Click “Add content type” to go to the content type edit form **A**.

Most of this form is familiar to you, because it looks somewhat like the one you use to create and edit nodes, which you learned about in Chapter 2, “Establishing Your Drupal Site.”

But there’s an important difference. Changes you make on a *node* edit form affect only a single node; changes you make on the *content type* edit form affect every node of that content type. You’re creating a *template* for future nodes.

3. Complete the two fields that give the content type its identity:
 - ▶ The Name field is where you type the “human-readable” name of the content type. Drupal automatically creates a “machine name” that converts spaces to underscores, and uppercase characters to lowercase characters. (Drupal uses this machine name internally.) However, you can choose to change the machine name by clicking the Edit link in this area **B**.
 - ▶ The optional Description field is where you add explanatory text that appears on the “Add new content” page to help content editors select the right content type.

continues on next page

4. Complete the “Submission form settings” section, which controls the appearance of several parts of the node edit form for this content type:

- ▶ The “Title field label” field comes with the default value of *Title*. I like to change it to something more descriptive, such as *Model name* **C**. You can’t leave this field blank.
- ▶ The “Preview before submitting” setting determines whether node creators see a Preview button at the bottom of the node edit form. If Disabled is selected, they see only the Save button; if Optional is selected, they see both the Save and Preview buttons; and if Required is selected, they see only the Preview button and are allowed to save a node only after previewing it.
- ▶ The “Explanation or submission guidelines” field lets you add a note that appears at the top of the node edit form for the benefit of node creators and editors **D**.

C Settings that affect the node edit form, with some options changed for our “Sunader Fans” site

D A portion of the node edit form, reflecting changes we made

E Publishing options settings on the content type edit form

F The display settings options for a content type

G The title and beginning of a node as it appears when the “Display author and date information” check box is selected

5. Click the “Publishing options” tab **E** to set default values for that part of the node edit form. To learn about these settings, see Chapter 3, “Creating and Managing Content.”

6. Click the “Display settings” tab **F** to choose whether a node shows its author and date of creation **G**, taken from the “Authoring information” section of the node edit form. Exactly *how* this information displays is defined by programming in the active theme.

continues on next page

7. Click the “Comment settings” tab **H** to control how visitors can “talk back” in response to nodes:

- ▶ The “Default comment setting for new content” pop-up menu lets you set whether comments will be allowed at all on nodes of this content type. (You can change this setting on individual nodes, however.)

Open means comments are allowed.

Closed prevents further comments from being added, but displays any that were already there.

Hidden means all existing comments will be hidden, and no more will be allowed.

- ▶ Threading affects whether text is indented so that conversations are easier to follow **I**. When not selected, all comments appear at the same indentation level. This “flat” format is graphically clean but (sometimes) makes it hard to follow individual conversations, which are known as “threads” **J**.
- ▶ The “Comments per page” setting defines how many comments appear before the reader has to click links at the bottom of the page to see more. A low value makes comment pages display more quickly, because there’s less information for Drupal to put together and for the server to push through the Internet. I generally don’t find the delay too onerous, so I usually change it to the maximum value (300).

H Changing the comment settings to affect visitors’ interaction with a content type

I Comments displayed as a threaded list, making it clear that the third comment is in response to the first

1986 Adventure-II

published by admin on Tue, 10/06/2010 - 13:52

The 19-foot Adventure-II holds a special place in the hearts of collectors.

Comments

admin
Tue, 10/06/2010 - 13:54
[permalink](#)

Just for the record...

I have a Sunrader Adventure-II, and I love it! :)

[reply](#)

janice
Tue, 10/06/2010 - 13:55
[permalink](#)

You do? You lucky!

You have one, admin? Wow! Where did you ever find it?

[reply](#)

Jim Bob
Tue, 10/06/2010 - 13:56
[permalink](#)

Yeah, tell us!

Like janice, I've been trying to get one of these for "forever".

Hey, admin, wanna sell yours? :)

[reply](#)

❶ Comments displayed as a flat (unthreaded) list, which is prettier but harder to follow

- ▶ The “Allow comment title” check box determines whether commenters can write their own subject lines. When unselected, or if a commenter fails to provide a title, Drupal automatically creates one using the first few words of the comment. For example, for a comment that reads “I agree very strongly with your post,” Drupal will generate the (somewhat awkward) subject line “I agree very strongly with.”
- ▶ The “Show reply form on the same page as comments” check box determines whether visitors see a comment form directly below the node’s content or have to click an “Add new comment” link to see it. Your choice makes a difference in how many comments you get; it’s surprising how many people won’t bother to click! To encourage *more* comments, leave this check box selected.
- ▶ “Preview comment” options work very much like those for a node’s “Preview post” options. When you select Required, commenters are forced to read through what they wrote before being allowed to post. In practice, selecting Required means that some commenters will fail to realize that they have to preview their posts before saving them and will simply lose their comments by going to another screen. On the other hand, some administrators believe the Required selection cuts down on frivolous, misformatted, and hot-headed comments.

continues on next page

8. Click the “Menu settings” tab **K** to control where you can put menu links to nodes of this content type when you create or edit nodes. See the section “To create a menu item that links to a node” in Chapter 3 to understand how your choice here affects users. You can always add links from any menu by directly editing it: For details, see Chapter 6, “Improving Access to Content.”
9. When you’ve made all necessary changes, click “Save content type.”

The screenshot shows the 'Menu settings' tab for a content type. On the left, a sidebar lists configuration sections: 'Submission form settings', 'Publishing options', 'Display settings', 'Comment settings', and 'Menu settings' (which is active). The main content area is divided into two columns. The left column contains a 'Menu settings' section with a text input field currently showing '<Main menu>'. The right column is titled 'Available menus' and lists 'Main menu' (checked), 'Management', 'Navigation', and 'User menu'. Below this list is a note: 'The menus available to place links in for this content type.' At the bottom of the right column is a 'Default parent item' section with a dropdown menu set to '<Main menu>' and a note: 'Choose the menu item to be the default parent for a new link in the content authoring form.' At the very bottom of the form are two buttons: 'Save content type' and 'Delete content type'.

K Controlling where nodes of this content type can get automatic menu links

Deleting Content Types: Harder Than It Sounds

By now you might have noticed that it seems as easy to delete a content type as it is to create one. Easier, in fact. You can delete them on both the content type edit form and the page that lists content types.

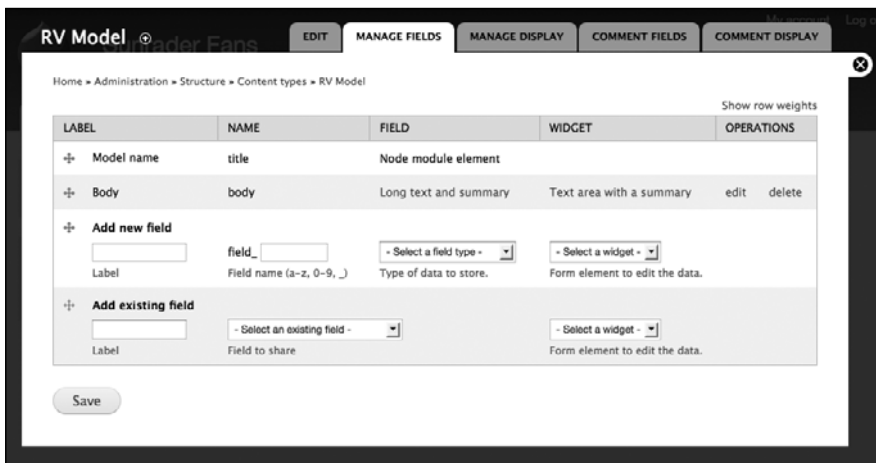
But beware: Nodes of a content type become “orphans” when you delete that content type, available for view but not editing. Only sophisticated monkeying around in the database will make them work again.

Also note that content types that are created by modules, such as “blog entry,” don’t have “delete” links in the usual places. Instead, you get rid of them by turning off the relevant module. Nodes of such content types also become orphaned when you do this, so be sure to delete them as you learned in the section “Finding, Editing, and Deleting Content” in Chapter 3. You can then re-create those nodes as a different content type. (Unfortunately, Drupal doesn’t currently offer any way to change a node’s content type.)

To change a field's display and input options:

1. In the Toolbar, click Structure and then click “Content types.”
2. Click the “manage fields” link next to the content type you want to affect. Besides being a place to add fields, on this screen **L** you can also:
 - ▶ Delete a field by clicking its “delete” link, if it’s a type of field that can be deleted. However, you can’t delete fields that modules create or the node’s title field (shown in **L** by the name we gave it earlier, “Model name”).
 - ▶ Edit the field’s basic criteria by clicking the “edit” link. We discuss the many options on the resulting page in the next step.
 - ▶ Change the order of fields by clicking and dragging their **+** icons. Note that you have to click Save after reordering fields: You’ll lose all reordering changes if you click other links on this page before clicking Save.
 - ▶ Change a field’s type, for example from List to Long text, by clicking the link that shows the current type. For the Body field in **L**, that’s “Long text and summary.” (It’s rare that you’ll want to do this.)
 - ▶ Change a field’s “widget”—that is, the selection tool for entering data in the field. Examples of widgets included in Drupal include check boxes, radio buttons, pop-up menus, and text fields; contributed modules may add other widgets (such as a date selection calendar). In **L**, the widget is currently “Text area with a summary.”

continues on next page



L The screen where you manage a content type’s fields

3. Click the “edit” link to go to the field settings form.

Options on this form vary considerably, depending on the field’s type and widget. (Our example is of the Integer field type.) In fact, some field types and widgets break these options into more than one screen, in which case you’ll simply have to fill out the first screen, then click “Save field settings” before moving on to the second.

The screen shown in **M** is for a newly created field that’s intended to hold an integer. (You’ll learn how to add fields in the next section, “To add fields to a custom content type.”) The form’s top part is for changing the field’s basic information, including:

- ▶ The Label, which you entered when you first created the field.
- ▶ The “Required field” check box, which determines whether to force node editors to enter a value in this field.
- ▶ The “Help text,” which will appear near the field as a guide for node editors.
- ▶ The Minimum, Maximum, Prefix, and Suffix allow you to limit the range of values the user can enter, or to change how the field appears when displayed. For example, you could put a dollar sign before the value so a “5” appears as “\$5.”

There are many other options, depending on the field type and widget selected. For example, the “Long text and summary” field type includes a “Summary input” check box that lets you permit (or forbid) entry of a summary section. For more about summaries, see the “Gaining More Control of Individual Nodes” section in Chapter 3.

The screenshot shows the 'Model Year' field settings form. The top navigation bar includes links for 'FIELD SETTINGS', 'FIELD TYPE', and 'DEFAULTS'. The main content area is titled 'Updated field Model Year field settings.' and contains two sections. The first section, 'FIELD SETTINGS', includes a 'Label' field with the value 'Model Year', a 'Required field' checkbox, 'Minimum' and 'Maximum' value fields, a 'Help text' area, and 'Prefix' and 'Suffix' fields. The second section, 'MODEL YEAR FIELD SETTINGS', includes a 'Number of values' field with the value '1' and a 'Maximum number of values users can enter for this field' field. A 'Save settings' button is located at the bottom of the form.

M The field edit form, where you change a field’s basic information

4. Fill out the “Default value” parts of the field edit form. Whatever data you enter here will appear prefilled in the field when someone later creates a node that contains it.
5. Complete the “Number of values” pop-up menu that allows you to decide how many values you can have in a multiple value field.

Multi-value fields can be a hard concept to grasp, and it’s best explained with a few examples. Such a field might be used to show:

- ▶ The names of gold, silver, and bronze medalists in a content type for Olympic sports events. (In that case, you’d set the “Number of values” pop-up menu to 3.)
 - ▶ The instruments that a musician plays in an “orchestra member” content type (with the pop-up value set to Unlimited).
6. When you’ve finished modifying the field’s settings, click “Save settings.”


To add fields to a custom content type:

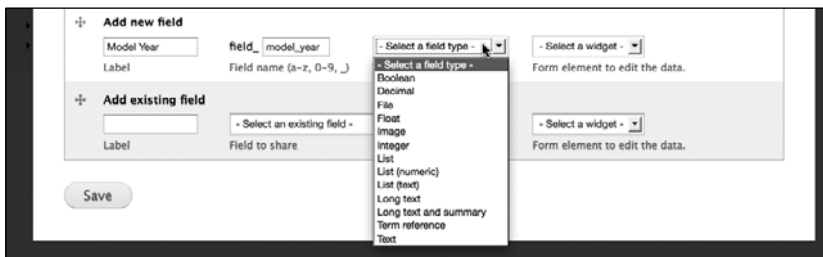
1. Click Structure in the Toolbar and then click “Content types.”
2. Click the “manage fields” link next to the content type you want to change.
3. Put the human-readable name in the Label area and the machine-readable name in the “Field name” area. Select a field type, which determines the kind of data this field can contain.

Generally speaking, Drupal's field types **N** fall into five categories:

- ▶ Text, including Text, Long text, and Long text and summary
- ▶ Numbers, including Integer, Float, Decimal, and Boolean
- ▶ Selection lists
- ▶ Files
- ▶ References, for example Term reference. (Downloadable modules permit references to other entities, such as nodes and users.)

Other field types may become available as you add modules, as Chapter 9 describes.

4. Once you've indicated the type of data this field will contain, the widget pop-up menu becomes available and reflects the widgets for the selected field type. Choose the one you want and then click Save.
5. The “Field settings” screen displays, where you specify details that are specific to the field type you chose. This section is broken into two screens: The first affects this field in every content type where it appears, while the second is specific to this content type. Complete the field settings form as was described in the section, “To change a field's display and input options” earlier in this chapter. When you've provided all required details, click “Save field settings.” You will return to the list of fields for this content type.
6. Put fields in the order you want by clicking and dragging their  icons. Click Save when finished.



N Types available for custom fields in a content type

Putting Images and Styled Text in Content

Drupal creator Dries Buytaert gives a “State of Drupal” keynote presentation at every semi-annual DrupalCon conference (drupalcon.org) to talk about progress during the last six months and plans for the next six. At that talk he also presents the results from his web poll on buytaert.net that asks, “What improvements does Drupal need most?” Lately, the same item has been at the top of Drupal developers’ wish list: better media handling.

With Drupal 7, that wish finally came true—at least, for graphics. In previous versions, you had two options. You could upload the graphic to the server, figure out where it was being stored, and write some HTML to put the graphic where you wanted; or you could find, download, and figure out a half-dozen modules to speed the process.

Under the guidance of Dries and long-time developer Angela Byron (known on the Drupal.org site as “webchick”), Drupal 7’s developers made a special effort to incorporate those modules into Drupal itself. The results aren’t perfect, but they’re definitely encouraging. As you learned in Chapter 3, you can now attach graphics to articles with the click of a button; this chapter tells you how to attach graphics to nodes of *any* content type and how to place graphics with greater precision than Drupal naturally allows. Finally, you’ll learn how to add text styles such as bold and italics without needing to resort to HTML code.

To add an image to article nodes:

1. To add an image to an existing article node, go to the node and click the Edit tab. Otherwise, create the article node as you learned to do in Chapters 2 and 3.
2. Complete the Title and Body fields as usual and make all desired changes to the settings tabs at the bottom of the page.
3. Click the Browse button next to the Image field to get a file-selection dialog box. Navigate to the graphics file you want to add and select it.
4. Click the Upload button. Depending on your connection speed, you might have to wait a while until the file uploads to your server. When it's finished, you'll see a small (thumbnail) version of the image, and the Upload button will become labeled Remove.
5. **Optional:** Add a description of the photo in the "Alternate text" field. This text will be useful to those visitors, such as blind people, who use non-graphical web browsers **A**.
6. At the bottom of the node-creation form, click Save to save the node. You'll view the newly changed article, with your full-size graphic at the top **B**.



A The node-creation form after adding an image and some alternate text



B An article with an added image

C Adding an image field to a content type

D The “field settings” screen

To enable images in nodes of any content type:

1. Go to the list of content types by clicking Structure in the Toolbar and then clicking “Content types.”
2. Click the “manage fields” link next to the content type you want to affect.
3. In the “Add new field” area, enter the human-readable and machine-readable names for the image field you’re going to add, as was described in the section “To add fields to a custom content type.”
4. In the area labeled “Type of data to store,” change the pop-up menu to Image. The widget pop-up menu to its right should also change to Image automatically C.
5. Click Save. You now see the “Field settings” screen D, which offers two choices:
 - “Upload destination” allows you to choose whether to make the images accessible to everybody who can find the URL (Public files) or only to those people who have special permission (Private files). It’s an advanced subject, and rare that you would want to specify the latter. Suffice it to say that if you don’t have a specific reason to select “Private files,” leave this selection on its default, which is “Public files.” (On some Drupal installations, “Public files” is the only option.)
 - “Default image” lets you select an image that will appear in this field for every node of this content type that doesn’t have its own image.
6. Click “Save field settings” to arrive at the settings page for this field, which we’ll discuss in the next section, “To control image criteria.”

To control image criteria:

1. When you create an image field, you eventually come to a second screen full of settings for that field.

To change these settings *after* a field has been created, you need to return to that second field settings screen. To do so, first go to the “Content types” list screen by clicking Structure in the Toolbar and then clicking “Content types.” Click “manage fields” next to the content type that contains the field you want to affect. Then click the “edit” link next to the relevant field **E**.

In either case, you’ll see the settings page for this image field **F**.

2. We described what many of these fields are in the section “To change a field’s display and input options.” Some of the options that are especially relevant to images include:

- ▶ “Allowed file extensions” lets you specify which file types are permissible in this field. The defaults (**png, gif, jpg, jpeg**) cover most graphic types that can be displayed in web browsers, but you might want to add other types (such as **svg**) for special applications.
- ▶ “Maximum image resolution” and “Minimum image resolution” let you ensure that graphics aren’t too big or small. People will still be able to upload larger graphics—Drupal will simply resize them down to fit these values. However, your site will reject attempts to upload graphics that are smaller than the minimum size. (If Drupal attempted to resize too-small graphics, the result would be an ugly, low-quality image.)



- E** Reaching the second field settings page from the list of fields

- F** The second field settings screen for images

- ▶ “Maximum upload size” lets you limit how much disk space each image may occupy. If you don’t enter anything, the limit will be defined by settings in the **php.ini** file in your AMP stack. For more about this, see Chapter 1, “Getting Drupal Up and Running.”
 - ▶ “Enable *Alt* field” and “Enable *Title* field” check boxes allow anyone who adds an image to also include text that appears in the HTML code that surrounds that image. As the help text explains, the *alt* field appears in places where the image itself can’t be loaded, such as in non-graphical web browsers; the *title* field determines what appears when a visitor hovers the cursor over the image.
3. Click “Save settings” to return to the list of fields for this content type. The changes you made will affect all nodes of this content type that you edit or create, but they won’t affect existing nodes.

TIP It’s a good idea to set the first value (which defines the width) in the “Maximum image resolution” field to 800 or less. If you don’t, people will upload graphics that are wider than visitors’ screens. When visitors look at the node, that graphic will force them to scroll horizontally to see the whole thing, and it could make your site’s design ugly and hard to read.

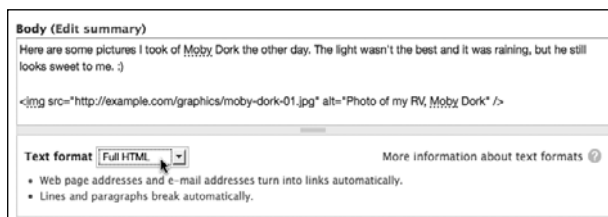
TIP If you leave the “Maximum upload size” setting blank, you’ll probably allow people to upload very big files, which could quickly fill your hard drive and will probably appear inconveniently large in your site. A value of 2 MB or 4 MB should be sufficient for most purposes.

TIP Enabling the Alt and Title fields is good practice, because it improves your site’s accessibility for people with limited vision and those browsing the web with unusual devices.

To include images in content using HTML:

1. Find and copy the URL of the image you want to use. One way of doing this is to go to the web page that contains it, put your pointer over the image, right-click (or Control-click if you have a one-button mouse), and select Copy Image Location.
2. Edit the node where you want to place an image by going to it and clicking the Edit tab.
3. In the node's Body field, type the HTML code ``, replacing *URL* with the web address you copied. You can style the image further using additional HTML, if you like.
4. Choose Full HTML from the “Text format” pop-up menu **G**.

Warning: Switching to the Full HTML text format opens a security hole if anyone else has permission to edit this node. For more information about text formats, see Chapter 5, “Making Content Interactive.”



G Displaying a remotely hosted image in the body of a node

What About Other HTML Tags?

The act of changing the text format from Filtered HTML to Full HTML opens the door to any and all HTML stylings, including text styles such as **** (bold) and ****.

However, your HTML might not appear as you expect. Drupal themes format pages using an extensive set of CSS directives, which can substantially change the appearance of even the most basic HTML tags.

5. Scroll to the bottom of the screen and click Save. The image now appears in the body of the node.

TIP To link to files that exist elsewhere on your Drupal site, you don't need the `http://domain-name` part of the URL. In fact, including that part slows down image retrieval, since your Drupal site is forced to look up its own address! A typical code in that case would be ``.

TIP I recommend that you reference only graphics that are hosted on your own server. For one thing, you have more control over the images and know they won't suddenly change or disappear; for another, it's considered impolite (and possibly illegal) to "hotlink" graphics that are hosted elsewhere, unless you have explicit permission from their owners, because the remote server pays the bandwidth charges for the image to appear on your web site.

By the same token: Once a file is on your server, you can display it in any node, not just the one to which it's attached.

To style text using a rich-text editor:

1. In a web browser, go to the Wysiwyg *project* page at drupal.org/project/wysiwyg.
2. Download and install the latest Drupal 7 version of the Wysiwyg module, using the techniques described in the “To install, enable, and configure modules” section of Chapter 9.
3. Click Modules in the Toolbar.
4. Scroll to the bottom of the page. Select the Wysiwyg check box and then click “Save configuration” **H**.
5. This module doesn’t actually add rich-text tools to your site. Instead, it connects your Drupal site to many rich-text editors that are available in the wider web-development world. You now need to download and install one of those packages.

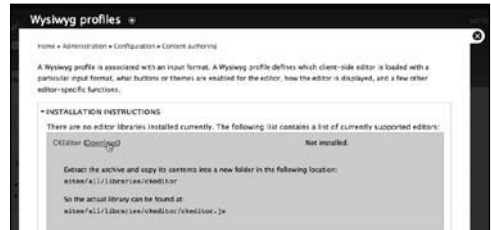
Click Configuration in the Toolbar and then click “Wysiwyg profiles.”

6. On the resulting page you’ll see a list of rich-text editors that work with the Wysiwyg module. You’ll need to download and install one of them: I’ll demonstrate using the first one on the list, the popular CKEditor.

Click the Download link **I** to go to the CKEditor site **J**. (You’ll need to return to this page for the instructions it contains, so I recommend that you open the CKEditor site in a new browser window or tab.)



- H** Enabling the Wysiwyg module



- I** The link to download the CKEditor package



- J** The CKEditor home page with the pointer indicating the package’s download link



K Selecting CKEditor for all times when users enter text in the Filtered HTML format

7. You'll now download and unpack CKEditor.

- On Mac or Windows: Click the "Download zip" button. Find the downloaded package on your computer and double-click it to uncompress it. (If you have problems doing so on a Windows computer, see the sidebar "Help! I Can't Uncompress the File!" in Chapter 1.)
- On *nix using the command-line interface: Copy the URL linked from the "Download tar.gz" button. In a terminal program, type **wget URL**. When the package has finished downloading, type **tar -xvzf downloaded-package-name**.

8. Return to the Wysiwyg profiles page to determine where you'll need to put the files you just downloaded and uncompress. Move the files as directed.

Since we're using CKEditor, we'll need to create a new "libraries" directory inside **/sites/all/modules**.

9. Reload the Wysiwyg profiles page, either using your browser's Reload function or by clicking Configure in the Toolbar and then clicking "Wysiwyg profiles."

The page has changed to now allow you to select which editor to use for each text format that's active on your site. (For more information about text formats, see Chapter 5.) You could also load other rich-text editors by clicking the "Installation instructions" link to expose the instructions you saw on this page earlier.

10. In the pop-up menu next to "Filtered HTML," select CKEditor **K** and then click Save.

continues on next page

11. The rich-text editor now appears whenever you have an opportunity to enter Filtered HTML text, for example when creating a basic page **L**.

TIP As you might guess from the long list of rich-text editors on the Wysiwyg configuration page, there are lots of debates in the web-development community about the advantages and disadvantages of each. Links from the Wysiwyg project page at drupal.org/project/wysiwyg lead you to those discussions and additional resources for understanding this complex topic.

TIP You can direct your site to use the rich-text editor when entering text in any text format, but I generally use it only for Filtered HTML. When I enter something in Plain Text format I usually don't want any rich-text formatting at all, while I reserve Full HTML format for times when I want to explicitly provide formatting in code.



L Editing a node with CKEditor's controls active

Index

A

Acquia Drupal

- contents, 13
- DAMP stack installer, 3, 8
- downloading, 13
- user-interface audit, 170
- administration emails, 161–162
- Administration module, 203
- administrative overlay, 22
 - turning off, 25
 - viewing pages with/without, 24

Advanced Help module, 204

Aggregator module, 61, 63–66

- Feed aggregator page, 63–66

all folder in **sites** folder, 39

AMP stacks

- DAMP, 3, 8
- elements
 - Apache web server, 2–3
 - MySQL database program, 2–3
 - PHP programming language, 2–3
- installing both stack and Drupal, 3

AMP stacks (*continued*)

MAMP

- Apache Ports preferences, 10
- configuring, 9–10
- DocumentRoot** folder, 10
- installing, 4–5
- MySQL databases, creating, 15
- MySQL Ports preferences, 10
- Web Sharing conflicts, 9–10

WAMP, 2

- configuring, 11
- DocumentRoot** folder, 11
- installing, 6–7
- MySQL databases, creating, 16
- PHP file size settings, 11

WampServer, 2

Anello, Michael, 205

Anonymous users, 154

Apache web server, 2–3

- localhost, 20

- Ports preferences (MAMP), 10

appropos copy *nix command, 13

appropos description *nix command, 13

- articles, 29, 31
 - versus* basic pages, 52
 - versus* blog posts, 62
- elements, 52
- images, 86
- tags, 54

Atom, 66

B

- backing up sites
 - databases, 45
 - installation files, 44
 - restoring from backups, 46–47
- Bartik theme, 22, 35
- basic pages
 - versus* articles, 52
 - versus* blog posts, 62
 - elements, 52
 - home (front), 29–30
- Basic theme, 183
- Berry, Addison, 211
- BigDump program, 47
- blocks, 140
 - adding, 145
 - configuring location and appearance, 143–145
 - moving among/within regions, 141–142
- blog posts, 61–62
 - versus* articles or basic pages, 62
- book pages, 61
 - of linked content, 68–69
 - structure, changing, 70
- Boulton, Mark, 170
- Buytaert, Dries, 13, 211
 - DrupalCon conference, 85
 - entities introduction, 173
- Byron, Angela, 85, 155

C

- CAPTCHA module, 204
- CCK (Content Construction Kit) module, 74, 203
- cd ..** *nix command, 12
- cd *directory name*** *nix command, 12
- cd */path/to/location*** *nix command, 12
- ChatZilla, 219
- Colloquy, Mac IRC client, 219
- command --help** *nix command, 13
- comments
 - Comment module, enabling, 99
 - responses to nodes, 58
- contact forms, 174–176
- containers, 101–102
- Content Construction Kit (CCK) module, 74, 203
- content/content types. *See also* custom content/content types; interactive content/content types
 - articles, 29, 31
 - versus* basic pages, 52
 - versus* blog posts, 62
 - elements, 52
 - images, 86
 - basic pages, 31
 - versus* articles, 52
 - versus* blog posts, 62
 - elements, 52
 - home (front), 29–30
 - blog posts, 31, 61–62
 - versus* articles or basic pages, 62
 - book pages, 61
 - of linked content, 68–69
 - structure, changing, 70
 - deleting, 71, 80
 - editing, 71–72
 - filtering, 72
 - finding, 71–72
 - image styles, 117
 - changing, 118

content/content types, image styles (*continued*)

- creating, 119–120

- editing, 119–120

images

- adding new, 87

- adding to existing nodes, 86

- adding with HTML, 90–91

- in modules, 203–204

- settings, 88–89

list of content, viewing, 71–72

news feeds, 63–67

- finding feeds through searches, 66

nodes

- appearing at top of pages, 60

- author and date, changing, 59

- backing up revisions, 56

- “Create new revision” check box, 56

- editing, 54, 71–72

- hiding, 59

- menu items linking to, 55

- new, 53

- page bodies, 52

- versus* pages, 31

- performing changes on multiple, 72

- tracking changes, 56

- URL paths, user-friendly, 57

- user comments, 58

- viewing list, 71–72

searching

- granting permissions, 123

- indexing content, 123, 126

- making content searchable, 122

- Search settings page, 126–127

- simple and advanced, 124–125

text formats

- adding, 112–115

- changing default, 111

- with HTML, Filtered HTML, 91, 110–112, 115

- with HTML, Full HTML, 91, 110–112, 115

core themes, 34

cp *nix command, 13

cron program, 123, 126

- Poormanscron module, 204

cropping images, 119–120

crowdsourcing, 171

CSS (Cascading Style Sheets)

- Skinr module, 146

- themes, 185–186

- adding styles to, 186–187

- external styles, 184

- graphics, background, 189–191

- graphics, incidental, 191–192

- importance, 187

- inline styles, 184

- internal styles, 184

CSS3: *Visual QuickStart Guide*, 187

custom content/content types, 74. *See also*

- content/content types; interactive

- content/content types

- deleting, 80

- fields

- adding, 84

- CCK (Content Construction Kit) module,
74, 203

- display and input options, 81–83

- new, 75–80

Cyberduck, 11

D

DAMP, 3, 8

dashboard, 23

databases. *See* MySQL databases

Date module, 203

default folder in **sites** folder, 39

desaturating images, 119

DocumentRoot folder

- MAMP, 10

- WAMP, 11

Douglass, Robert, 127

- Drupal Foundation, 226
- DrupalCamps, 228
- DrupalCon conferences, 85, 228
- Drupal/Drupal 7
 - administrative overlay, 22
 - turning off, 25
 - viewing pages with/without, 24
- contributing, 225
 - non-technical support, 226
 - technical support, 227
- Dashboard, 23
 - customizing, 27
- downloading, 8
- face-to-face opportunities
 - Drupal classes, 228
 - DrupalCamps, 228
 - Drupalcon, 228
 - DUGs (Drupal User Groups), 228
- finding employment, 224
- glossary, 229–233
- hiring experts, 220
 - checking out at other sources, 223–224
- installing, 17–18
 - configuring databases, 18–19
 - configuring site, 19–20
 - Drupal 7 and AMP stack, 3
 - fix of MAMP problem, 5
- modules
 - top 20 used, 203–204
 - version problems, 197, 202
- *nix servers
 - transferring to, 11–12
 - unpacking on, 12
- requirements
 - AMP stack, 2–3
 - file transfer program, 2
 - Internet connection, 2
 - local computer *versus* server, 3
 - text editor, 2

- Drupal/Drupal 7 (*continued*)
 - Shortcut bar, 22
 - changing items in, 26
 - Toolbar, 22
- Drupal/Drupal 7 community, building and protecting
 - contact forms, 174–176
 - profiles
 - fields, adding, 172–173
 - Profile module, 173
- drupal.org** web site
 - discussion etiquette, 217
 - help
 - asking questions, 216
 - faceted searches, 214–215
 - handbooks, 213
 - modules, 216
 - Site Building Guide, 213
 - themes, 216
 - Troubleshooting FAQ, 213, 216
 - user accounts, 213–214
- DUGs (Drupal User Groups), 228

E – F

- email addresses, 149, 158
 - administration, automated, 161–162
- etiquette, discussions, 217
- Feed aggregator page, 63–64
- file transfer protocol programs. *See* FTP
- FileField module, 203
- FileZilla, 2, 11
- Filtered HTML text format, 91, 110–112, 115
- Firebug, 187, 191
- Firefox browser, 187
 - ChatZilla, 219
- Flexinode module, 74
- “45 Modules in 45 Minutes,” 205
- “49 Modules You Should Know,” 205

forums

- Comment and Taxonomy modules, 99
 - creating, 102
 - help, 216
 - managing, 103
 - multi-level structure, 101
 - setting up, 99–100
- Forums vocabulary, 104, 105
- front home) pages, 29–30
- promoting nodes to, 60
- front pages. *See* home (front) pages
- FTP (file transfer protocol) programs
- requirement, 2
 - transferring Drupal 7 to *nix servers, 11–12
- Full HTML text format, 91, 110–112, 115, 116
- Fusion theming, 146

G

- Garland theme, 35
- Genesis theme, 183
- global settings, themes, 37
- glossary, 229–233
- GNU Image Manipulation Program (GIMP), 189
- GNU Public License, 180
- Lullacons Pack 1, 192
 - modules, 197
- Google Analytics module, 204
- graphics files, themes, 33. *See also* images
- backgrounds, 189–191
 - incidental, 191–192
- gzip files, uncompressing, 8

H

- Haug, Nate, 192
- help
- drupal.org** web site
 - asking questions, 216
 - faceted searches, 214–215

help, **Drupal.org** web site (*continued*)

- handbooks, 213
 - hiring experts, 220
 - modules, 216
 - one-on-one with IRC, 218–219
 - Site Building Guide, 213
 - themes, 216
 - Troubleshooting FAQ, 213, 216
- face-to-face opportunities
- Drupal classes, 228
 - DrupalCamps, 228
 - Drupalcon, 228
 - DUGs (Drupal User Groups), 228
- hierarchical arrangement, menus, 136–137
- home (front) pages, 29–30
- promoting nodes to, 60
- .htaccess** hidden file, 46

I

- Image module, 204
- ImageAPI module, 203
- ImageCache module, 203
- ImageField module, 203
- images. *See also* graphics files, themes
- adding
 - to existing nodes, 86
 - with HTML, 90–91
 - new, 87
 - modules
 - Image, 204
 - ImageAPI, 203
 - ImageCache, 203
 - ImageField, 203
 - settings, 88–89
 - styles, 117
 - changing, 118
 - creating, 119–120
 - editing, 119–120
 - user accounts, 152

- IMCE module, 204
- indexing content
 - cron program, 123, 126
 - periodically, 123
- .info files, themes, 33
 - CSS files, 184–186
 - duplicating/renaming, 181–183
 - graphics files, 189, 192
- INSTALL.TXT file, 28
- interactive content/content types. *See also* content/content types; custom content/content types
- containers, 101–102
- forums
 - Comment and Taxonomy modules, 99
 - creating, 102
 - managing, 103
 - multi-level structure, 101
 - setting up, 99–100
- polls
 - creating, 96–97
 - managing, 98
- taxonomies, categorizing with, 104
- taxonomy vocabulary
 - planning on paper, 109
 - setting up, 105
 - terms, adding, 108–109
 - terms, viewing in content, 109
 - using, 106–107
- internationalization, 162
- Internet connection, requirement, 2
- ISPs (Internet service providers), MySQL tools access, 16

J – L

- jQuery UI module, 204
- layout
 - blocks, 140
 - adding, 145

- layout, blocks (*continued*)
 - configuring location and appearance, 143–145
 - moving among/within regions, 141–142
 - panels, 146
- license agreements
 - GNU Public License, 180
 - core themes, 180
 - Lullacons Pack 1, 192
 - modules, 197
 - MAMP, 4–5
- links to menus
 - to administration page, 131–133
 - deleting, 134
 - disabling Secondary, 139
 - in Main and User menus, 138
 - to nodes, 55, 131–133
 - URL required, 137
- Linux operating system, *nix servers, 3
- list of content, viewing, 71–72
- local computers *versus* servers, Drupal development, 3
- locale settings, 153
- localhost, 20
- ls -al** *nix command, 12–13
- ls** *nix command, 12
- Lullacons Pack 1, 192

M

- Macintosh
 - Colloquy IRC client, 219
 - DAMP, lack of support, 3
 - Drupal 7
 - downloading, 8
 - hidden file, 8
 - MAMP, 2
 - Apache Ports preferences, 10
 - configuring, 9–10
 - DocumentRoot** folder, 10

Macintosh, MAMP (*continued*)

- installing, 4–5

- MySQL databases, creating, 15

- MySQL Ports preferences, 10

- Web Sharing conflicts, 9–10

- TextEdit, 2

Main menu, 128–129

- changing link locations, 138

- Delete button, lack of, 134

- disabling, 139

Maintenance mode, 48

MAMP, 2

- Apache, Ports preferences, 10

- configuring, 9–10

- DocumentRoot** folder, 10

- fix for Drupal installation problem, 5

- installing, 4–5

MySQL

- creating databases, 15

- Ports preferences, 10

- superuser password, 5

- Web Sharing conflicts, 9–10

man *command* *nix command, 13

Management menu, 128–129

- Delete button, lack of, 134

menus

- deleting, 134

links

- to administration page, 131–133

- deleting, 134

- disabling Secondary, 139

- in Main and User menus, 138

- to nodes, 55, 131–133

- URL required, 137

Main menu, 128–129

- changing link locations, 138

- Delete button, lack of, 134

- disabling, 139

Management menu, 128–129

- Delete button, lack of, 134

menus (*continued*)

- Navigation menu, 128–129

- Delete button, lack of, 134

- new, 55, 129–130

- rearranging, 135

- in hierarchy, 136–137

- themes vary appearances, 137

- titles, changing, 137

- User menu, 128–129

- changing link locations, 138

- Delete button, lack of, 134

- metadata text files, themes, 33

- Microsoft Internet explorer fix, 185

- Miranda, Windows IRC client, 219

modules

- administrative overlay, 22

- turning off, 25

- available outside **drupal.org** web site, 197

- configuring, 195–196

- enabling, 195

- evaluating

- “Comparison of contributed modules”
page, 209

- past, through issue queues, 206–209

- reading blogs, 205

- Similar Module Review group, 209

images

- Image, 204

- ImageAPI, 203

- ImageCache, 203

- ImageField, 203

- installing, 194–195

- popular, top 20, 203–204

- turning on, 28

- updating, 198–200

- bug fixes, 201

- improvements/additions, 201

- security fixes, 201

- version availability, 197, 202

- modules** folder, 39

monitoring sites

system activity, tracking, 41–42

system issues, correcting, 40

mv source destination *nix command, 12

MySQL databases, 2–3

case sensitivity, 15

creating with phpMyAdmin, 14

Mac OS X using MAMP, 15

*nix with command-line interface, 16

Windows using WAMP, 16

naming restrictions, 15

Ports preferences (MAMP), 10

superuser password

for MAMP installation, 5

for WAMP installation, 7

updating, 199–200

N

Navigation menu, 128–129

Delete button, lack of, 134

news feeds, 63–67

finding feeds through searches, 66

NineSixty theme, 183

*nix servers, 7

commands, 12–13

DAMP, 3, 8

Drupal 7

database with command-line interface, 16

transferring to *nix, 11–12

unpacking on *nix, 12

XAMPP stack, security warning, 7

nodes

appearing at top of pages, 60

author and date, changing, 59

backing up revisions, 56

“Create new revision” check box, 56

current region, 31

editing, 54

hiding, 59

nodes (*continued*)

menu items linking to, 55

new, 53

versus page content types, 31

performing changes on multiple nodes, 72

promoting to home (front) pages, 60

tracking changes, 56

URL paths, user-friendly, 57

user comments, 58

viewing list, 71–72

NoneOne series, modules, 205

Notepad (Windows), 2

O – Q

Overlay module, 22

pages. *See* content/content types; custom
content/content types; interactive
content/content types

panels, 146

passwords, 149

Pathauto module, 203

Permission filter, viewing user accounts, 151

permissions

defining, 163

editing, 164, 168

finding settings for specific modules, 169

important permissions, 170

moderating comment spam, 177

Photoshop, 189

PHP programming language, 2–3

file size settings in WAMP, 11

PHP filter, text formats, 110, 113, 115–116

phpMyAdmin, MySQL databases

backing up, 45

creating, 14

importing/exporting problems, 47

for Mac OS X using MAMP, 15

for *nix using command-line interface, 16

restoring, 46–47

for Windows using WAMP, 16

Planet Drupal, 205

polls

creating, 96–97

managing, 98

Poormanscron module, 204

profiles

fields, adding, 172–173

Profile module, 173

programming files, themes, 33

pwd *nix command, 12–13

R

RDF (Resource Description Framework), 66

README.TXT file, 28

rearranging, 135

requirements, Dupral 7

AMP stack

Apache web server, 2–3

MySQL database program, 2–3

PHP programming language, 2–3

file transfer program, 2

Internet connection, 2

local computer *versus* server, 3

text editor, 2

Role filter, viewing user accounts, 151

roles, 149, 163

adding/deleting, 164

administrative users, 164–166

anonymous users, 164–166

changing, for individuals, 166

rotating images, 119–120

RSS (Really Simple Syndication), 66

S

scaling images, 120

SCP (Secure Copy), 2

script files, themes, 33

Secure Copy (SCP), 2

servers *versus* local computers, Drupal development, 3

Seven theme, 20, 22

Shortcut bar

changing items in, 26

new in Drupal 7, 22

signatures and avatars, 159–160

Site Building Guide, 213

sites folder, 44

all and **default** folders, 39

themes, changing existing, 183

Skinnr module, 146

slogans, themes, 31

Status filter, viewing user accounts, 151

style files, themes, 33

T

Tags vocabulary, 104

tags/tagging, 54

taxonomies

categorizing with, 104

vocabulary

planning on paper, 109

setting up, 105

terms, adding, 108–109

terms, viewing in content, 109

using, 106–107

Taxonomy module, 99

Teague, Jason Cranford, 187

text editor, requirement, 2

text formats

adding, 112–115

changing default, 111

with HTML, Filtered HTML, 91, 110–112, 115

with HTML, Full HTML, 91, 110–112, 115, 116

with HTML tags, 91

htmlLawed module, 110

with PHP filter, 110, 113, 115–116

with rich-tech editor, 92–94

selecting for individual nodes, 111

TextEdit (Mac), 2

themes

- administration, 22–23

- alternative, downloading and installing, 38–39

- Bartik, 22, 35

- Basic, 183

- core, 34

- CSS (Cascading Style Sheets), 185–186

 - adding styles to, 186–187

 - external styles, 184

 - graphics, 185

 - graphics, background, 189–191

 - graphics, incidental, 191–192

 - importance, 187

 - inline styles, 184

 - internal styles, 184

- duplicating, 181–185

- Garland, 35, 185

- Genesis, 183

- menu appearances, 137

- new, 180

- NineSixty, 183

- parts, 33

- renaming, 181–184

 - versus* changing existing theme, 185

- settings, 35–37

 - global, 37

- Seven, 20, 22

- slogans, 31

- template files, 185

- Zen, 183

themes folder, 39

Token module, 197, 203

Toolbar

- changing items in, 139

- new in Drupal 7, 22

TopNotchThemes, 146

Troubleshooting FAQ, 213, 216

U

uncompressing gzip files, 8

Unix: Visual QuickStart Guide, 4th Edition, 13

Unix/Unix-like operating systems, *nix servers, 2–3

updating sites, 48–50

user accounts

- automated administration emails, 161–162

- canceling, 154–155

 - controlling, 156–158

- creating, 148–150

 - controlling, 156–158

- email addresses, 149, 158

- internationalization, 162

- locale settings, 153

- modifying, 152–153

- notifying of new account, 149

- passwords, 149

- versus* people terminology, 150

- permissions

 - contact forms, 174–176

 - defining, 163

 - editing, 164, 168

 - finding settings for specific modules, 169

 - important permissions, 170

 - moderating comment spam, 177

- pictures, 152

- profiles

 - fields, adding, 172–173

 - Profile module, 173

- roles, 149

 - adding/deleting, 164

 - administrative users, 164–166

 - anonymous users, 164–166

 - changing, for individuals, 166

 - comment spam, 177

 - contact forms, 163, 174

- signatures and avatars, 159–160

user accounts (*continued*)

status, 149

user names, 148

viewing list of, 151

User menu, 128–129

changing link locations, 138

Delete button, lack of, 134

user names

defining, 148

versus people terminology, 150

V – W

Views module, 203

WAMP, 2

configuring, 11

DocumentRoot folder, 11

PHP file size settings, 11

installing, 6–7

MySQL

databases, creating, 16

superuser password, 7

WAMP Setup Wizard, 6

WampServer, 2

list at http://en.wikipedia.org/wiki/Comparison_of_WAMPs, 2

Web Sharing (Mac) conflicts, 9–10

web site management

backing up

databases, 45

installation files, 44

restoring from backups, 46–47

configuring information, 31–32

monitoring

system activity, tracking, 41–42

system issues, correcting, 40

moving to another computer, 47

sites folder, **all** and **default** folders, 39

updating, 48–50

web sites, helpful information

accounts, <http://domain-name/admin/config/people/accounts>, 36

Acquia Drupal, acquia.com/downloads, 3, 13

AMP stacks

acquia.com/downloads, 3

apachefriends.org/en/xampp.html, 7

list at http://en.wikipedia.org/wiki/Comparison_of_WAMPs, 2

mamp.info, 2, 4

sourceforge.net, 6

wampserver.com/en/, 2, 6

BigDump program, <http://ozerov.de/bigdump.php>, 47

blog posts

<http://domain-name/blog>, 62

<http://domain-name/blog/user-ID>, 62

contributing

drupal.org/contribute, 224

drupal.org/contribute/development, 227

drupal.org/contribute/documentation, 226

drupal.org/contribute/marketing, 226

drupal.org/contribute/testing, 226

drupal.org/contribute/translations, 226

<http://lists.drupal.org/listinfo/support>, 226

CSS (Cascading Style Sheets)

w3schools.com/css, 187

w3schools.com/css/css_background.asp, 189

databases, <http://drupal.org/node/81995>, 47

Drupal Foundation, <http://association.drupal.org>, 226

Drupal/Drupal 7

cyrve.com/d7cx, 202

drupal.org, 13

drupal.org/node/43816, 39

web sites, Drupal/Drupal 7 (*continued*)

- drupal.org/project/drupal, 8
- drupal.org/project/issues/project-name, 217
- drupal.org/project/usage, 202
- drupal.org/Troubleshooting-FAQ, 213
- <http://domain-name/update.php>, 49
- [/sites/default/settings.php](http://sites/default/settings.php), 47
- events, <http://groups.drupal.org/events>, 228
- Firebug, getfirebug.com, 187, 191
- forums, <http://domain-name/forum>, 101
- FTP
 - cyberduck.ch, 11
 - filezilla-project.org, 2, 11
 - winscp.net, 11
- GNU Public License, gnu.org/copyleft/gpl.html, 180
- help
 - drupal.org/forum, 216
 - drupal.org/Troubleshooting-FAQ, 213
- internationalization, <http://groups.drupal.org/i18n>, 162
- IRC connections
 - drupal.org/irc, 218
 - <http://webchat.freenode.net>, 218
- job hunting, groups.drupal.org/jobs, 224
- menu links, drupal.org/project/views, 131
- modules
 - drupalmodules.com, 197
 - drupal.org/project/modules, 28, 193
 - drupal.org/project/token, 197
 - <http://domain-name/article-about-bears#overlay=admin/modules>, 25
 - <http://localhost/admin/modules>, 25
- modules, evaluating
 - drupal.org/node/266179, 209
 - drupal.org/project/issues/module-name, 206, 208
 - drupal.org/project/issues/statistics/module-name, 207
 - drupal.org/project/module-name, 216
 - groups.drupal.org/similar-module-review, 209
 - nodeone.se/blogg/49-modules-you-should-know, 205
 - MySQL superuser password, udopage.com/developers/setting-up-passwords-for-wamp.php, 7
 - new accounts, <http://domainname/user/register>, 156
 - news feeds, savemyhomebook.com, 66
 - Node 8, webchick.net/contributor-spotlight/daniel-kudwien, 155
 - panels, drupal.org/project/panels, 146
 - search options
 - <http://domain-name/search>, 124
 - lullabot.com/articles/drupals_search_module_and_scoring_factors, 127
 - text formats
 - drupal.org/handbook/customization/php-snippets, 116
 - drupal.org/project/htmlawed, 110
 - themes
 - drupal.org/project/basic, 183
 - drupal.org/project/genesis, 183
 - drupal.org/project/ninesixty, 183
 - drupal.org/project/themes, 34, 38, 180–181
 - drupal.org/project/zen, 183
 - drupal.org/theme-guide, 179, 184
 - fusiondrupalthemes.com, 146
 - topnotchthemes.com, 146
 - uncompressing gzip files
 - stuffit.com/win-expander.html, 8
 - technelysium.com.au/winimp.html, 8
 - winace.com, 8
 - winzip.com, 8
 - Webform module, 204
 - Weitzman, Moshe, 202

Windows

Drupal 7

- downloading, 8

- uncompressing gzip files, 8

Miranda IRC client, 219

Notepad, 2

WAMP, 2

- configuring, 11

- DocumentRoot** folder, 11

- installing, 6–7

- MySQL databases, creating, 16

- PHP file size settings, 11

WinSCP, 11

X – Z

XAMPP, 7

Zen theme, 183