



From Technologies to Solutions

Drupal 5 Views Recipes

94 recipes to develop custom content displays for your
Drupal web site

Marjorie Roswell

[PACKT]
PUBLISHING

Drupal 5 Views Recipes

94 recipes to develop custom content displays for your
Drupal web site

Marjorie Roswell



BIRMINGHAM - MUMBAI

Drupal 5 Views Recipes

Copyright © 2009 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, Packt Publishing, nor its dealers or 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 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: May 2009

Production Reference: 1200509

Published by Packt Publishing Ltd.
32 Lincoln Road
Olton
Birmingham, B27 6PA, UK.

ISBN 978-1-847196-96-5

www.packtpub.com

Cover Image by Vinayak Chittar (vinayak.chittar@gmail.com)

Credits

Author

Marjorie Roswell

Reviewers

Dave Myburgh

Aaron Stewart

Acquisition Editor

David Barnes

Development Editor

Ved Prakash Jha

Technical Editors

Bhupali Khule

Hithesh Uchil

Indexer

Monica Ajmera

**Production Editorial
Manager**

Abhijeet Deobhakta

Editorial Team Leader

Akshara Aware

Project Team Leader

Lata Basantani

Project Coordinator

Leena Purkait

Proofreader

Jeff Orloff

Production Coordinator

Aparna Bhagat

Cover Work

Aparna Bhagat

About the Author

Marjorie Roswell has been building web sites that serve the community for more than a decade.

She developed a GIS system for assisting citizen callers to the Baltimore Office of Recycling, and has taught professional classes in desktop publishing, AutoCAD, and Drupal. She has served non-profit clients at Community IT Innovators, and now designs web sites and maps with Roswell Infographics.

At Connecticut College, Marjorie received the Betty Flanders Thomson Prize for Excellence in Botany. She combines her love of plants and Drupal to create <http://BaltimoreUrbanAg.org> for the Baltimore Urban Agriculture Task Force.

Marjorie enjoys writing and playing music with a community of local musicians. You can hear some of her music at <http://FriendlyCoffeehouse.org>.

Acknowledgement

I have lots of people to happily and gratefully thank.

Thanks to my parents, Arthur and Betty Roswell, for always expressing interest in something I'm not sure they ever learned to pronounce.

Thanks to Bob Roswell, my computer scientist brother. He gave me my first *real job* at ComputerLand back in 1985, and set me up with my Linux web server about a decade later.

Thanks to Barbara Roswell, my sister-in-law, and a professor of writing, for thoughtful and helpful ideas.

Thanks to my sister, Judy Roswell Weinstein and her wonderful family who managed to get me out on a glacier when I thought I should be writing during vacation. Her oldest son, Ari (AriX, at <http://drupal.org/user/214187>) was a Drupal GHOP participant, and his family usability study yielded a few installation patches in Drupal 6.

Thanks to Earl Miles (merlinofchaos, at <http://drupal.org/user/26979>) for writing the amazing Views module, and to Daniel F. Kudwien (sun, at <http://drupal.org/user/54136>), for co-maintaining Views 5.x.

Thanks to the awesome world Drupal community. (As I write this, the newest members of drupal.org come from Uganda, Belgium, United States, India, Hungary, Ireland, and Finland.) The book appendices list the maintainers of hundreds of Views-enabled modules.

Thanks to my nearest Drupal neighbor, mathematician Emil Volcheck (<http://groups.drupal.org/user/33466>), for kindly testing out the Node Reference recipes.

Thanks to Aaron Stewart for preparing a helpful script to download all versions of Drupal 5 modules.

Thanks to FreeDigitalPhotos.net for the lovely photographs used in our gallery, lightbox2, and carousel recipes. (Photo sources are detailed in the book downloads in Chapter 6, *Recipe 59*)

Thanks to Curtis Clark for creating the terrific Pie Charts for Maps font.

Thanks to Rob Ellis and James Harris, of MySafeWork.com, for the site screenshot and their good work.

Thanks to Joe Sances for permission to use his poster, *Unarmed Truth*, in a screenshot.

Thanks to Peter Montague for free reign to share content from <http://rachel.org>. Dr. Montague introduced me to the Precautionary Principle, and compelling information on health and the environment.

Thanks to Stephanie Pakrul, for permission to adapt one of her beautiful Top Notch Themes (<http://topnotchthemes.com>).

Thanks to Leena Purkait, Project Coordinator, for keeping track of innumerable chapters, rewrites, worksheets, and appendices, and for always including smiley faces when inviting me to submit chapters on time.

Thanks to Ved Prakash Jha, Development Editor, for reading and thoughtfully responding to every single one of my emails!

Thanks to David Barnes, Senior Acquisition Editor for liking my book idea enough to send a contract.

Thanks to Reviewer Dave Myburgh, (ncrn8, at <http://drupal.org/user/51467>) whose eagle eye has greatly improved this book, and whose positive remarks in the document margins warmed the heart of the author.

Thanks to my Technical Editors, Hithesh Uchil and Bhupali Khule, who have made this a better book.

Kudos to the Friendly Coffeehouse community, the Baltimore Urban Agriculture Task Force, and Baltimore EarthSave. Every one of them does such important work and play that each merited a Drupal website.

Thanks to many friends: Leah Ulansey, Naomi Bernstein, Matthew Clark, Joanne Stato, Jonathan Rudie, and Mark Edwards, who contributed the lovely music for the podcasting recipe.

Thanks to Wordsmith Joe for words and support.

Thanks to friend and client, Dan Walsh, who has more than once given me the chance to say, "I've got a recipe for that!"

About the Reviewer

Dave has been involved with computers since before the web existed. He studied as a molecular biologist, but discovered that he liked working with computers more than bacteria. He had his own computer business in South Africa (where he grew up) doing technical support and sales. He even created a few static web sites for clients during that time.

He went back to science for a few years when he first came to Canada, and then got sucked into the world of Drupal when a friend wanted a site for a local historical society. Since then he has once again started his own company that now builds websites exclusively in Drupal (he doesn't *do static* anymore). There is no lack of work in the Drupal world and he now balances his time between work and family. He has also reviewed several Drupal books including Drupal 5 Themes, Drupal 6 Themes.

I would like to thank my family for being so supportive of me and what I do. Working from home can be a mixed blessing sometimes, but having the opportunity to watch my son grow up makes it all worthwhile.

Table of Contents

Preface	1
Chapter 1: Introduction to Views	7
Recipe 1: Installing the Views module	7
Finding or creating the <DRUPALROOT>/sites/all/modules directory	8
Downloading and uncompressing the module	8
Enabling the module	10
Recipe notes	11
Recipe 2: Views-related URLs on your site	13
Recipe notes	13
Recipe 3: Creating a "Swim Groups" Page View	14
Creating a new Content type (Swim Group), and adding content	14
Creating a View	17
Selecting Filters	19
Recipe notes	22
Recipe 4: Adding a Header to your View	23
Recipe notes	24
Recipe 5: Adding a View to a Menu	25
Recipe notes	27
Recipe 6: Creating a Block of Swim Groups	27
Recipe notes	29
Recipe 7: Creating an "Admin" Role and setting Access Permissions	30
Installing the Admin Role module	30
Adding an Admin Role	31
Configuring the Admin Role module	31
Creating a User for the Admin Role	32
Recipe notes	32
Recipe 8: Installing Firefox and Firebug	32
Installing Firefox	32
Installing Firebug	32
Recipe notes	35
Recipe 9: Revealing the full extent of Views	35
Ensure that Firefox and Firebug are installed (See Recipe 8)	36

Running the jQuery command	37
Recipe notes	37
Recipe 10: Preparing Views Worksheets	38
Recipe notes	38
Summary	40
Chapter 2: Working with Default Views	41
Recipe 11: Adding "Recent Comments" to your site	43
Overriding the View	45
Configuring the Block	45
Configuring the Date Field	47
Recipe notes	47
Recipe 12: The frontpage View	49
Enabling the Views RSS module	50
Looking at the Frontpage View, and Overriding it	50
Basic Information fieldset	51
Page fieldset	51
Fields fieldset	51
Feed Selector Argument	51
Filters fieldset	52
Sort Criteria fieldset	53
Enabling the frontpage View to show on the front page	54
Removing duplicate search engine results	55
Recipe notes	55
Recipe 13: Read More link	56
Recipe notes	57
Recipe 14: All-time Popular, and Recent Popular Pages	57
Enabling Statistics	58
Overriding the Default Views	58
The Menu Options	59
Recipe notes	62
Recipe 15: All-time Popular, and Recent popular Blocks—removing the hit count	63
Adding the blocks to a region	63
Removing the hits from the popular_alltime block	64
Recipe notes	65
Another alternative	66
User-submitted snippets	66
Recipe 16: Taxonomy View	67
Recipe notes	70
Recipe 17: Fixing the case of the missing Taxonomy term	72
Recipe notes	74
Recipe 18: Empty Text, with PHP	74
Generating a list of Regional categories	76
Recipe notes	77
Recipe 19: Getting to know the Tracker	78
Recipe notes	81

Recipe 20: Making the Tracker Sortable	82
Recipe notes	83
Recipe 21: Integrating the Tracker View with the Tracker module	84
Recipe notes	85
Recipe 22: Reviewing Default Views code	85
Recipe notes	88
Recipe 23: Creating a Default View	90
The ".info" file	90
Default Views function in a ".inc" file	90
Including the Default View in a ".module" file.	92
Recipe notes	93
Summary	94
Chapter 3: CCK and Views	95
Recipe 24: Addressing the Views Body Field issue	95
Demonstrating the Body Field issue	96
Copying the Body Field, and modifying the View	98
Recipe notes	99
Recipe 25: Creating Website snapshots	100
Installing modules	100
Creating a Content Type	100
Adding the Link Field	101
Create a Website Snapshots View	101
Recipe notes	103
Recipe 26: Creating a Blogroll	105
Modifying the Link Field	106
Creating a new view based on website_snapshots	106
Recipe notes	107
Recipe 27: Displaying Guitar Chords	108
Recipe notes	109
Recipe 28: Using the Viewfield module	110
Installing modules	110
Views, Taxonomy, and Content setup	110
Associating the View with a Content item	112
Recipe notes	113
Recipe 29: Using the Node Reference module	115
Enabling the modules	115
Creating the Media Hit and Press Release Content types	115
Adding Press Releases	116
Creating drop-down Press Release View	117
Modifying and populating the Media Hit Content type	117
Recipe notes	118
Recipe 30: Relating Content types with Reverse Node Reference	118
Recipe notes	120
Recipe 31: Embedding a View in a Theme template	121
Recipe notes	126

Recipe 32: Displaying mini-pie charts	126
Setting up the Pie Chart Content type and View	127
Install and configure sIFR	130
Recipe notes	132
Summary	133
Chapter 4: Dates and Calendars	135
Recipe 33: Understanding Date formats	135
Recipe notes	140
Recipe 34: Block of upcoming workshops	140
Recipe notes	143
Recipe 35: Exposed Filters	144
Recipe notes	145
Recipe 36: Views Date Range Filter module	145
Recipe notes	146
Recipe 37: Exposed Filter settings	146
Recipe notes	148
Recipe 38: Summary Views	148
Recipe notes	151
Recipe 39: Date Browser	151
Recipe notes	153
Recipe 40: Timeline	153
Recipe notes	156
Recipe 41: Views Popup	157
Recipe notes	159
Recipe 42: Creating a Calendar	159
Installation and configuration	160
View setup	160
Installing Calendar Blocks	162
Recipe notes	163
Recipe 43: iCal feeds	164
Import Calendar items	165
Export Calendar items	166
Recipe notes	167
Recipe 44: Upgrade Date and Calendar modules	168
Backing up the Database	168
Upgrading Calendar and Date	170
Recipe notes	174
Summary	174
Chapter 5: Views and Tools for Administrators	175
Recipe 45: Administration menu	175
Recipe notes	177
Recipe 46: ModuleInfo	177
Recipe notes	178
Recipe 47: Views UI permissions	178
Recipe notes	181

Recipe 48: Views UI Sort	181
Recipe notes	183
Recipe 49: Editablefields	183
Recipe notes	184
Recipe 50: Views Bulk Operations—Content administration	184
Recipe notes	188
Recipe 51: Views Bulk Operations—Taxonomy assignment	189
Recipe notes	194
Recipe 52: Views Custom Field	195
Recipe notes	196
Recipe 53: Cron setup	196
Example 1: Setting up Cron using cPanel	197
Example 2: Setting up Cron on a Linux box	199
Example 3: Using the Poormanscron module	200
Recipe notes	200
Recipe 54: Formatting Views queries	201
Recipe notes	203
Summary	204
Chapter 6: Views Galore	205
Recipe 55: Views Fusion	205
Patch the Node Reference module	206
Views setup	206
Views Fusion installation and configuration	207
Recipe notes	209
Recipe 56: Views Fast Search	209
Setup	209
Using the Default View	209
Using an Alternative Imported View	210
Replacing the Search block	211
Recipe notes	212
Recipe 57: YouTube Video Bar	212
Recipe notes	214
Recipe 58: Views Bonus Pack Export	215
Recipe notes	217
Recipe 59: Photo Gallery	218
Set up the Photo Content type	218
Configure ImageCache	219
Add Photos	221
View setup	221
Recipe notes	222
Recipe 60: Gallery with Lightbox2 and jQuery Update	223
Installing jQuery Update	223
Installing Lightbox2	223
Recipe notes	224
Recipe 61: Google Maps	225
Setup Google Maps Tools	225

Create and populate the Content types	226
View setup	228
Recipe notes	230
Recipe 62: Proximity Search	232
Recipe notes	238
Recipe 63: Views Calculations	239
Recipe 64: Podcasting	242
Setup	242
Adding content	243
View setup	245
Recipe notes	247
Recipe 65: Using the Flag module for Bookmarks	249
Default Bookmarks View	249
Recipe 66: Using the Flag module for various use cases	250
Viewing a List of inappropriate comments	253
Creating a view for Fact-Checkers	253
View of RSVPs	254
Recipe notes	256
Recipe 67: Explore more Views, Galore	257
Views Checkboxes	257
Phoneblogz	258
Default Views	258
Summary	259
Chapter 7: Theming and Layout	261
Recipe 68: Themeable functions reference	262
Recipe notes	263
Recipe 69: Displaying Themeable function names in Source Code	263
Recipe notes	264
Recipe 70: Using Panels with SimpleFeed	265
Setup	265
Creating Feed Content	266
Create Views	267
Enabling specific Views for Panels	268
Panels setup	268
Placing Panels on the Front Page	272
Recipe notes	272
Recipe 71: Tweaking output with hook_link_alter	273
Recipe notes	275
Recipe 72: Debugging	276
Recipe notes	278
Recipe 73: Node-feed-item.tpl.php	279
Recipe notes	280
Recipe 74: Format the Pager with CSS-only modifications	280
Recipe notes	282
Recipe 75: Overriding the theme_pager function	283

Recipe notes	284
Recipe 76: Understanding how Theme functions are called	285
Recipe notes	286
Recipe 77: Override theme_views_view	288
Recipe notes	290
Recipe 78: Total Item count	290
Recipe notes	291
Recipe 79: Views Theme Wizard	292
Recipe notes	296
Recipe 80: Date badge	298
Recipe notes	301
Recipe 81: Grouped View	302
Recipe notes	303
Recipe 82: Body classes	304
Recipe notes	305
Recipe 83: Carousel View	306
Recipe notes	308
Recipe 84: Views Dashboard—code approach	309
Recipe notes	311
Recipe 85: Views Dashboard—module approach	311
Recipe notes	313
Recipe 86: Great Online Theming resources	313
Summary	315
Chapter 8: Navigating the Online Drupal Community	317
Recipe 87: Creating an issue Page Bookmark	317
Recipe notes	318
Recipe 88: Searching the Views issue queue	319
Basic Search	319
Advanced Search	319
Recipe notes	320
Recipe 89: Posting an issue	321
Posting a New issue	321
Responding to an existing issue	322
Recipe notes	323
Recipe 90: Applying a Patch	323
Recipe notes	325
Recipe 91: Creating a Patch	326
Recipe notes	327
Recipe 92: Installing a Search plugin for the Contributions API	328
Recipe notes	329
Recipe 93: Using IRC—real-time technical support	330
Recipe notes	331
Recipe 94: Exploring Resources	332
Summary	333

Table of Contents

Appendix A: Default Views in Drupal 5 Modules	335
Appendix B: Formatters	343
Appendix C: Style Plugins	355
Appendix D: Views Hooks for Coders	361
Appendix E: Modules Included in Recipe Ingredients	363
Appendix F: Additional Resources and Modules Mentioned in Recipes	369
Appendix G: Selected Noteworthy Patches to Views	373
Index	379

Preface

The Views module is a powerful and flexible tool for displaying content on a Drupal site. Views came on the Drupal scene in 2005. Developed primarily by an awesome coder named Earl Miles with much input from other Drupalers, it has revolutionized the Drupal website developer experience.

This book offers step-by-step Views recipes for use by Drupal administrators and themers. We will be cooking up an extensive smorgasbord. We'll create lists, tables, calendars, charts, maps, proximity searches, fused views, views bulk operations, and so on, while introducing a wide array of techniques and companion modules along the way. Most of the recipes require no programming, but some contain useful code snippets.

This book covers Views for Drupal 5.x. (Views 2 will be the primary subject of a separate book.)

Why release a book on a Drupal 5 module, when Drupal 6 has already been released? It's simple: Many web site developers use Drupal 5!

There are often good reasons for this. Web site developers like to be sure that required modules are in good working shape for the version of Drupal they select. If contributed modules required for the site are not yet ported to Drupal 6, it is a reasonable option to use Drupal 5. Also, if a developer is replicating most of the functionality of an existing Drupal 5 site, she may use Drupal 5. Many developers *inherit* web sites that are still based on Drupal 5 code. Additionally, developers are still releasing new Drupal 5 modules. Drupal 5 remains a rich and widely-used framework.



What this book covers

Chapter 1 introduces the Views module along with a Views Worksheet that will serve as a guide throughout the rest of the book. We create a basic view of Swim Groups.

In *Chapter 2* we look at the six *default views* that come with the Views module. We learn which modules must be enabled for these views to appear and we get to know the feed selector and Term arguments. We end the chapter by creating a module with a default view of Swim Groups.

In *Chapter 3* we incorporate CCK fields into views such as a Blog Roll, Guitar Chord Display, and a Block of Media Hits Associated with Press Releases. We share a variety of ways to associate views with related content, including Viewfield and Node Reference modules and building views from within a template.

In *Chapter 4* we focus on date fields, calendars, upcoming events, and the timeline module. This chapter rounds out coverage of all of the Views User Interface fieldsets, including Exposed Filters.

In *Chapter 5* we examine ten tools to ease the work of site administrators, including Views Bulk Operations, Editable Fields, ModuleInfo, and setting up cron.

In *Chapter 6* we explore some rarely-covered but powerful modules, including: Views Fusion, Views Fast Search, Views Bonus Pack Export, and Google Maps Tools. We also create a YouTube Video Bar, detail the steps to install jQuery Update with a Lightbox2 Gallery, and examine how to use the Flag module for a variety of applications.

Chapter 7 is the theming chapter and a bit more code-rich than the others. We create a directory of available theme functions. We look closely at Views theme functions and override them when necessary. We cover some debugging strategies. We use CSS to create *date badges* and to control carousel formatting. Finally, we incorporate views into panels and tabs.

In *Chapter 8* we focus on navigating the online Drupal community. We learn how to search the Views issue queue, and how to apply and create patches. We cover some Dos and don'ts for Drupal chat on IRC and create a few helpful browser buttons and search plug-ins to facilitate finding information online.

Appendix A lists all the default views available for Drupal 5

Appendix B gives a comprehensive list of Drupal 5 field formatters by module

Appendix C gives a Comprehensive list of Drupal 5 style plug-ins by module

Appendix D lists the Views 1 hooks

Appendix E lists the modules included in recipe ingredients

Appendix F lists the additional resources and modules

Appendix G lists the selected noteworthy patches to views, sorted by topic

What you need for this book

For most of the recipes, you'll need a Drupal 5 web site along with administrative access to the server and the site. (The Packt book, *Building Powerful and Robust websites with Drupal 5* offers a comprehensive chapter on installation.) You may also want to use a printer or copy machine to prepare copies of the Views Worksheet.

Who this book is for

This book is primarily written for Drupal site builders, administrators, and themers who want to develop custom content displays using Views. It can be used by anyone who has developed or inherited a Drupal 5 site.

Some knowledge of HTML and CSS is required. PHP basics will be handy for some of the recipes.

How you read this book is your choice. Some readers will start from Recipe 1, working all the way through to the Appendix, to gain as much information as possible. Others will head straight for the most-needed recipe, turning back only to complete prerequisite steps. However you approach it, cook up a storm!

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: "The last argument of that command was `views.module`."

A block of code will be set as follows:

```
if ($field['fullname'] == 'node_counter.totalcount') {
    $item .= ' <span class="popular-content-count">
(' . views_theme_field('views_handle_field', $field['queryname'],
$fields, $field,
    $node, $view) . ')</span>';
}
```

Any command-line input or output is written as follows:

```
cd <DRUPALROOT>/sites/all/modules/cck
```

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 our text like this: "Enable the core **Search** module and the **Views Fast Search** module".

 Warnings or important notes appear in a box like this. 

 Tips and tricks appear like this. 

Reader feedback

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

To send us general feedback, simply drop an email to feedback@packtpub.com, and mention the book title in the subject of your message.

If there is a book that you need and would like to see us publish, please send us a note in the **SUGGEST A TITLE** form on www.packtpub.com or email suggest@packtpub.com.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on www.packtpub.com/authors.

Customer support

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

Downloading the example code for the book

Visit http://www.packtpub.com/files/code/6965_Code.zip to directly download the example code.

The downloadable files contain instructions on how to use them.

Errata

Although we have taken every care to ensure the accuracy of our contents, mistakes do happen. If you find a mistake in one of our books – maybe a mistake in text or 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 to improve subsequent versions of this book. If you find any errata, please report them by visiting <http://www.packtpub.com/support>, selecting your book, clicking on the **let us know** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata added to any list of existing errata. 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 web site name immediately so that we can pursue a remedy.

Please contact us at copyright@packtpub.com with a link to the suspected pirated material.

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

Questions

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

1

Introduction to Views

In this chapter, we'll create both a page and a block using the **Views** module. We'll install Views, and introduce the **Basic Info**, **Page**, **Block**, **Fields**, and **Filters** fieldsets. We'll have a chance to further observe every available *Views UI fieldset* in *Recipe 9*. The *Views Worksheet* in *Recipe 10* summarizes the whole interface, and will help you track your learning throughout the book. While most chapters lend themselves to a pick-and-choose approach, I encourage you to complete each recipe in this chapter.

Let's Begin!

Recipe 1: Installing the Views module

Ingredients*

Drupal 5: <http://drupal.org>

Host login access

Views: <http://drupal.org/project/views>

Views UI: contained within the Views module

*These ingredients are prerequisite for all of the recipes that follow, and will not be listed in subsequent ingredient lists.

The **Views** module has a straightforward installation, much like other Drupal module installs. The following recipe instructions incorporate a mix of Windows tools and the command line for locating files, creating directories, downloading files, and unzipping them. Consider following the recipe, but feel free to substitute file management approaches specific to your operating system and preferences. This recipe is followed by a page listing helpful command line shortcuts that you should incorporate into your practice if they are available on your host.

Finding or creating the <DRUPALROOT>/sites/all/modules directory

1. Log in to your web server. You will need to know your host name, login name, and password.
2. Locate the Drupal root directory on your server (a common directory location is /home/your_account/public_html).
3. If this is the first contributed module in a brand new Drupal installation, you will need to create the modules subdirectory.
 - From your Drupal root directory, type the following to change into the <DRUPALROOT>/sites/all directory:
`cd sites/all`
 - Make a new modules directory:
`mkdir modules`
 - While you're there, you may as well create the theme directory, if it is not there already. We will use this directory in Chapter 7, *Techniques for Theming Views*
`mkdir themes`
 - Now change to the modules directory:
`cd modules`

Downloading and uncompressing the module

4. Open the browser of your choice.



Most of the screenshots in this book use the Firefox browser—a popular choice for many developers. *Recipe 8* covers Firefox installation.

Go to <http://drupal.org/project/views>, and read the page. There are several different versions of Views available for download. We are looking for the **Download** link marked **Recommended for 5.x**.

Releases				
Official releases	Date	Size	Links	Status
6.x-2.3	2009-Feb-07	1.3 MB	Download · Release notes	Recommended for 6.x ✓
5.x-1.6	2007-Jul-14	207.36 KB	Download · Release notes	Recommended for 5.x ✓
Development snapshots	Date	Size	Links	Status
6.x-2.x-dev	2009-Feb-26	1.31 MB	Download · Release notes	Development snapshot ✗
5.x-1.x-dev	2009-Feb-21	243.75 KB	Download · Release notes	Development snapshot ✗
View all releases 				

5. Download the module from the Views project web page. Here is one way to do that:

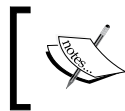
- Right-click in Firefox on **Download**.
- Select **Copy Link Location** (in Internet Explorer, this link will be called **Copy Shortcut**). You now have text similar to the following in your clipboard:

```
http://ftp.drupal.org/files/projects/views-5.x-1.6.tar.gz
```

A file with the `.tar.gz` extension is affectionately known as a "tarball", reflecting the days when files were backed up onto tape archives.

- Switch to your Drupal server window. Be sure you are still in your `<DRUPALROOT>/sites/all/modules` directory.
- Type:

```
wget <Paste><Enter>
```



In many shell environments, you can use the right mouse button to paste text from the clipboard. In Windows, you can use `Ctrl+V`.

- The result on your screen, after pasting the download link, will be:

```
wget http://ftp.drupal.org/files/projects/views-5.x-1.6.tar.gz
```

After pressing *Enter*, the download begins. You are provided with a status report of the percent complete as shown in the following screenshot:

```
PuTTY
infographics:/home/pingpong/public_html/sites/all # mkdir modules
infographics:/home/pingpong/public_html/sites/all # mkdir themes
infographics:/home/pingpong/public_html/sites/all # cd modules
infographics:/home/pingpong/public_html/sites/all/modules # wget http://ftp.drupal.org/
files/projects/views-5.x-1.6.tar.gz
--11:03:33--  http://ftp.drupal.org/files/projects/views-5.x-1.6.tar.gz
=> 'views-5.x-1.6.tar.gz'
Resolving ftp.drupal.org... 64.50.238.52, 64.50.236.52
Connecting to ftp.drupal.org|64.50.238.52|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 212,338 (207K) [application/x-gzip]

100%[=====>] 212,338      440.26K/s

11:03:33 (439.43 KB/s) - 'views-5.x-1.6.tar.gz' saved [212338/212338]

infographics:/home/pingpong/public_html/sites/all/modules #
```

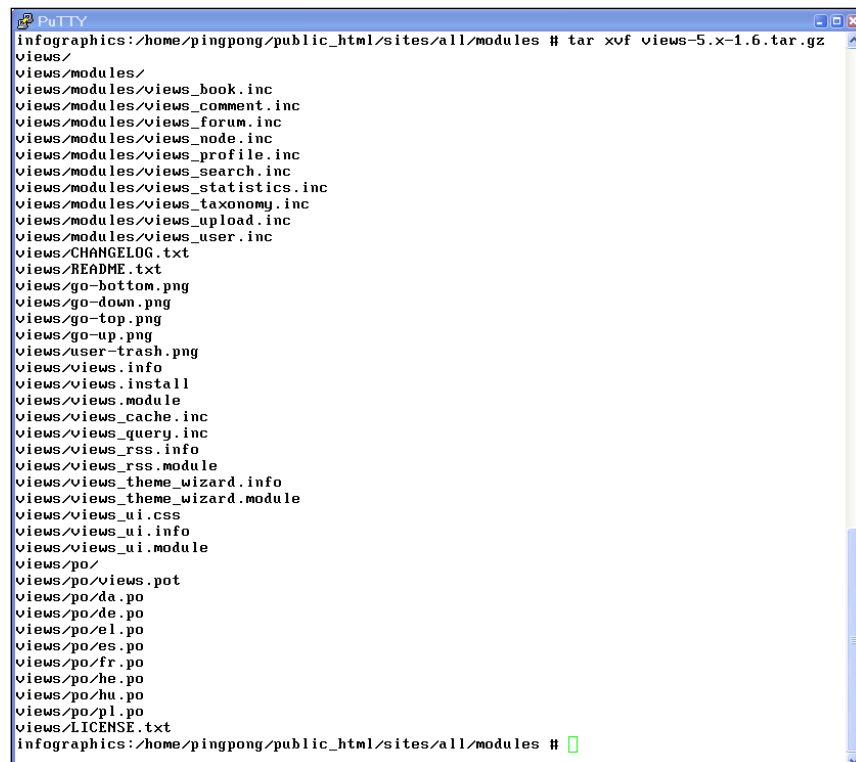
6. Uncompress the module to the `<DRUPALROOT>/sites/all/modules/views` directory. Enter:

```
tar xvf views-5.x-1.5.tar.gz
```

Depending on your system setup, you may need syntax such as this:

```
tar -zxvf views-5.x-1.5.tar.gz
```

You may also use an alternative uncompression tool, such as **7-Zip** or **WinRAR**. WinRAR offers a graphical interface in Windows, but is also available with a command line interface for Mac and Linux systems. The output of the `tar xvf views-5.x-1.5.tar.gz` command is seen in the following screenshot:

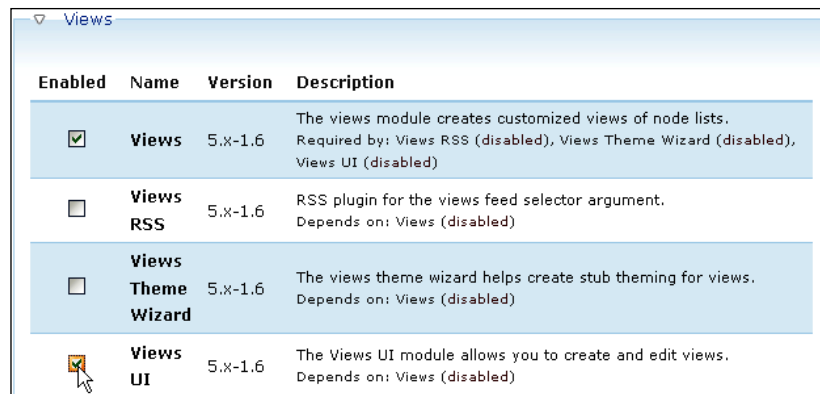
A screenshot of a PuTTY terminal window. The title bar says "PuTTY". The command prompt shows the user is in the directory `/home/pingpong/public_html/sites/all/modules` and has executed `tar xvf views-5.x-1.6.tar.gz`. The output lists the contents of the tar archive, including various .inc files, .txt files, .png files, .info files, .module files, .css files, .pot files, and .po files, as well as a LICENSE.txt file. The prompt ends with a green cursor.

```
infographics:/home/pingpong/public_html/sites/all/modules # tar xvf views-5.x-1.6.tar.gz
views/
views/modules/
views/modules/views_book.inc
views/modules/views_comment.inc
views/modules/views_forum.inc
views/modules/views_node.inc
views/modules/views_profile.inc
views/modules/views_search.inc
views/modules/views_statistics.inc
views/modules/views_taxonomy.inc
views/modules/views_upload.inc
views/modules/views_user.inc
views/CHANGELOG.txt
views/README.txt
views/go-bottom.png
views/go-down.png
views/go-top.png
views/go-up.png
views/user-trash.png
views/views.info
views/views.install
views/views.module
views/views_cache.inc
views/views_query.inc
views/views_rss.info
views/views_rss.module
views/views_theme_wizard.info
views/views_theme_wizard.module
views/views_ui.css
views/views_ui.info
views/views_ui.module
views/po/
views/po/views.pot
views/po/da.po
views/po/de.po
views/po/el.po
views/po/es.po
views/po/fr.po
views/po/he.po
views/po/hu.po
views/po/pl.po
views/LICENSE.txt
infographics:/home/pingpong/public_html/sites/all/modules #
```

Enabling the module

7. Return to your browser window. Go to the **Module** page at:
`http://YOURSITE.com/admin/build/modules`.

If you are not already logged into your Drupal site as an administrator, you will need to do so. Scroll down to the **Views** fieldset and enable the **Views** and the **Views UI** modules by clicking the two associated checkboxes.

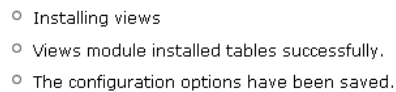


Enabled	Name	Version	Description
☒	Views	5.x-1.6	The views module creates customized views of node lists. Required by: Views RSS (disabled), Views Theme Wizard (disabled), Views UI (disabled)
☐	Views RSS	5.x-1.6	RSS plugin for the views feed selector argument. Depends on: Views (disabled)
☐	Views Theme Wizard	5.x-1.6	The views theme wizard helps create stub theming for views. Depends on: Views (disabled)
☒	Views UI	5.x-1.6	The Views UI module allows you to create and edit views. Depends on: Views (disabled)

- Click on the **Save configuration** button.



Congratulations! You will now see a message (similar to the screenshot that follows), signaling the successful installation of our module.

- 
- Installing views
 - Views module installed tables successfully.
 - The configuration options have been saved.

Recipe notes

UI is an abbreviation for "User Interface". The **Views UI** module offers a point-and-click approach for selecting content and output types, controlling view sort order, and so on. The **Views UI** can even generate a views programming code for you, as we'll see in Chapter 2. If the **Views UI** module were not enabled, you could still see the existing views and create new views through code. However, the **Views UI** module is quite helpful: even "Ninja" Drupal programmers will typically enable it.

Command line tips

These shortcuts will assist you in your Drupal development (your host environment may, or may not, have all of these shortcuts enabled).

- Press **Ctrl+U** in the command line to clear everything to the left of the cursor. For instance, if you paste something, and then realize the wrong text was in the clipboard, it may be easier to clear the whole line than to press backspace many times.

- Press **Ctrl+K** on the command line to clear everything to the right of the cursor.
- Press the up arrow to re-use a previous command line. You may also search your command history with **Ctrl+R**.
- Type **!\$** on the command line to reuse the last argument of the previous command line. For example:

```
more views_ui.module views.module
```

lets you page through the text of both modules. The last argument of that command was `views.module`. If your next command is:

```
less !$
```

this will print as:

```
less views.module
```

(This tip is difficult to find with a search engine. A Google search for **!\$** yields nothing.) Quit both the **more** and **less** commands by typing:

```
Q
```

- Use "man" pages. Man is shorthand for the manual.

```
man less
```

reveals that

Less is a program similar to more but which allows backward movement in the file as well as forward movement.

- Type **cd** to go to your home directory.

To return to the previous directory you were in, type:

```
cd -
```

- If you are not sure where you are in your directory structure, type:

```
pwd
```

This is the "print working directory" command, which provides the full path to your current directory. This is especially useful if your prompt is not set up to already display the full path.

- Use the **Tab** key for command line completion. In step 6 of the recipe above, you could type:

```
tar xvf v<tab>
```

If there is only one file that begins with `v` in the directory, the **Tab** key will conveniently complete the full name of the zipped `.tar` file:

```
tar xvf views-5.x-1.5.tar.gz
```

Press *Enter* to run the command.

If you already had the `votingapi` module installed, for instance, you would need to enter:

```
tar xvf vo<tab>
```

to distinguish **voting** from **views**.

Recipe 2: Views-related URLs on your site

Ingredients
Administrative access to a Drupal web site and a printer

This is a quick tour of Views-related URLs on your site. Make yourself at home. Feel free to click around and generally become familiar with the **Views** interface.

1. **Administer Views:** <http://YOURSITE.com/admin/build/views>
2. **Add a View:** <http://YOURSITE.com/admin/build/views/add>
(This is where we'll be spending much of our time in this book.)
3. **Import a View:** <http://YOURSITE.com/admin/build/views/import>
4. **Views Tools:** <http://YOURSITE.com/admin/build/views/tools>
5. **Views Help:** <http://YOURSITE.com/admin/help/views>
6. **Views UI Help:** http://YOURSITE.com/admin/help/views_ui
(While you are on this page, go ahead and print it. This is an easy page to miss, but quite informative.)
7. **Views Permissions:** <http://YOURSITE.com/admin/user/access#module-views>
(We will interact with this page in *Recipe 7*.)
8. **Enable Views modules:** <http://YOURSITE.com/admin/build/modules>

Recipe notes

- If your site does not have the **Clean URLs** option **Enabled**, you will need to add `?q=` before the URL parameters in the URL. For instance:

```
http://YOURSITE.com/admin/build/views
```

becomes:

```
http://YOURSITE.com/?q=admin/build/views
```

Recipe 3: Creating a "Swim Groups" Page View

[Ingredients]
Administrative access to a Drupal website

You may think of the Views UI as a series of questions. Here are some sample questions to consider:

- What do I want to name this view?
- What do I want to display?
- Who should be able to view this content?
- In what format should the content appear?
- Do I want the content to appear in a page, or in a block (or both)?
- How many items to display?
- Is there any particular order (By date? Alphabetically?)

We will first need to create the content for our view.

Creating a new Content type (Swim Group), and adding content

1. Go to **Administer | Content management | Content Types**. Select **Add content type** (<http://YOURSITE.com/admin/content/types/add>).

If you have the **Content Construction Kit (CCK)** module installed, you will see more than two tab options, but as seen in the following screenshot, we simply have the Drupal 5 core, plus the Views modules that we have installed.

Home > Administer > Content management

Content types

List [Add content type](#)

To create a new content type, enter the human-readable name, the machine-readable name, and all other relevant fields that are on this page. Once created, users of your site will be able to create posts that are instances of this content type.

Identification

Name: *

Swim Group

The human-readable name of this content type. This text will be displayed as part of the list on the *create content* page. It is recommended that this name begins with a capital letter and consists only of letters, numbers, and **spaces**. This name must be unique to this content type.

Type: *

swim_group

The machine-readable name of this content type. This text will be used for constructing the URL of the *create content* page for this content type. This name may consist of only of lowercase letters, numbers, and underscores. Dashes are not allowed. Underscores will be converted into dashes when constructing the URL of the *create content* page. This name must be unique to this content type.

Description:

Preschool, Youth, Teen, and Adult Swim Group

A brief description of this content type. This text will be displayed as part of the list on the *create content* page.

2. Enter **Name**, **Type**, and **Description**.
3. Keep the **Submission form** fieldset at default settings.
4. In the **Workflow** fieldset, uncheck **Promoted to front page** and set the **Default comment setting** to **Disabled**.

Workflow

Default options:

☒ Published

☐ Promoted to front page

☐ Sticky at top of lists

☐ Create new revision

Users with the *administer nodes* permission will be able to override these options.

Default comment setting:

☒ Disabled

☐ Read only

☐ Read/Write

Users with the *administer comments* permission will be able to override this setting.

5. Save your new content type. You will see a list of your **Content types**, and a message that your new content type has been added.

The screenshot shows the 'Content types' administration page in Drupal. At the top, there's a breadcrumb trail: 'Home > Administer > Content management'. Below this, the page title is 'Content types' with two links: 'List' (active) and 'Add content type'. A message box states: 'The content type *Swim Group* has been added.' Below the message is a table listing the content types on the site.

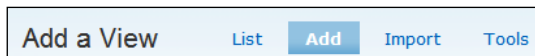
Name	Type	Description	Operations
Page	page	If you want to add a static page, like a contact page or an about page, use a page.	edit delete
Story	story	Stories are articles in their simplest form: they have a title, a teaser and a body, but can be extended by other modules. The teaser is part of the body too. Stories may be used as a personal blog or for news articles.	edit delete
Swim Group	swim_group	Preschool, Youth, Teen, and Adult Swim Group	edit delete

6. Go to **Create Content | Swim Group**, and add at least three content items (<http://YOURSITE.com/node/add/swim-group>).

The screenshot shows the 'Submit Swim Group' form in Drupal. At the top, there's a breadcrumb trail: 'Home > Create content'. The form title is 'Submit Swim Group'. It has a 'Title: *' field with the value 'Water Wrigglers (Ages 2 to 3)'. Below this is a 'Body:' field with the text: 'For those new to the water. Basic skills such as floating, kicking, and blowing bubbles. The group is led by professional child educators who are also certified water instructors'. There are expandable sections for 'Input format', 'Log message:', 'Menu settings', 'Comment settings', 'Authoring information', and 'Publishing options'. At the bottom, there are 'Preview' and 'Submit' buttons. A mouse cursor is pointing at the 'Submit' button.

Creating a View

7. Go to **Administer | Site Building | Views** (<http://YOURSITE.com/admin/build/views>), and select **Add**.



8. In the **Basic Information** fieldset, enter the following:

- **Name:** swim_group
- **Description:** List of swim groups

 A screenshot of the 'Basic Information' fieldset. It contains three sections: 'Name' with a text input field containing 'swim_group' and a description below it; 'Access' with two checkboxes, 'anonymous user' and 'authenticated user', both unchecked; and 'Description' with a text input field containing 'Listing for Swim Groups' and a description below it.

9. Sections of the main Views UI form are contained in HTML fieldsets. Click on **Page** to open that fieldset if it is not already expanded. Note that when a fieldset is closed, a triangle icon points to the fieldset name (as seen in the following image); when the fieldset is expanded, the triangle icon points down. This is a subtle clue (available in most themes) to help you navigate the page.



10. Enter the following values in their respective fields:

- ☒ **Provide Page View**
- **URL:** swim-groups

- **View Type: Full Nodes**
- **Title: Swim Groups**
- ☒ **Use Pager**
- **Nodes per Page: 20**

▼ Page

☒ Provide Page View

If checked this view will be provided as a page. If not checked, the fields in this group will be ignored.

URL:

Enter the URL to use for this view in the form of 'dir/dir'. Do not begin or end the URL with a /. Example: 'view/tracker'. This is required if providing a page view. You can also add \$arg as a placeholder for arguments passed in the URL, for example 'user/\$arg/tracker' or 'node/\$arg/related'. Note that any arguments listed here will be required, even if they are listed as optional below. You do not need to list arguments at the end of the path. Do not try to use URLs such as taxonomy/term/\$arg.

View Type:

▼

How the nodes should be displayed to the user.

Title:

The title that be shown at the top of the view. May be blank. This title ignores arguments; if you want your title to take arguments into account, use the "title" field in the arguments section.

☒ Use Pager

If checked this query may be multiple pages. If not checked this query will be one page.

☐ Breadcrumb trail should not include "Home"

If checked the breadcrumb trail for this page will discard "Home". Usually you will not set this, but this is used for the Front Page View, where it IS Home and should not leave a trail to itself.

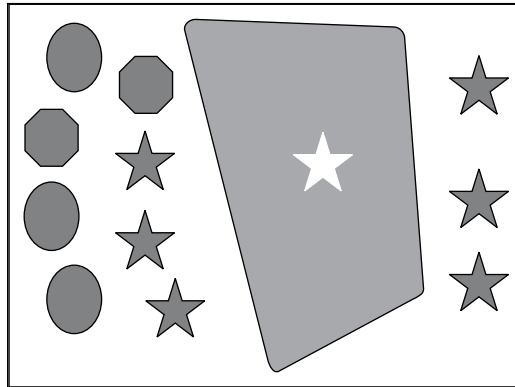
Nodes per Page:

The number of nodes to display per page. If 0, all nodes will be displayed. If not using a pager, this will be the maximum number of nodes in the list.

In this section, we are essentially creating a new **URL** on the site, and displaying full content items (nodes). We are specifying that up to **20** items will be displayed on the page, and if we have more, we will have the option to see additional items on subsequent pages.

Selecting Filters

The following figure offers a visual representation of Views filtering. We see the multiple content types available on the site symbolized on the left (for instance, Page, Story, Swim Group). The filter allows only one content type to be displayed. In our case, we just want the Swim Group.



11. Open the **Filters** fieldset. If you printed the **Views UI Help**, as suggested in *Recipe 2*, now is a good time to review the Filters options. There are many filter options available. We will select just two from the **Add Filter** drop-down list, **Node: Type**, and **Node: Published**. These are by far the two most commonly used filters. Select **Node: Type** from the drop-down, and click on **Add Filter**.

A screenshot of the 'Add Filter' fieldset. It features a light blue header with the text 'Add Filter'. Below the header is a light blue box containing a dropdown menu with 'Node: Type' selected and a small downward arrow. To the right of the dropdown is a gray button with the text 'Add Filter'.

The filter is added "above" the drop-down list.

Select the **Value** as **Swim Group** and leave the **Is One Of** option of **Operator**, as shown in the following screenshot:

The screenshot shows the 'Filters' section of a software interface. It contains a table with the following columns: Field, Operator, Value, Option, and Ops. The first row is configured with 'Node: Type' in the Field column, 'Is One Of' in the Operator column, and a dropdown menu in the Value column showing 'Page', 'Story', and 'Swim Group' (which is selected). There is an 'Expose' button and a small blue icon in the Ops column. Below the table, there is a text label 'Include or exclude nodes of the selected types.' and an 'Add Filter' section with a dropdown menu showing 'Comment: Pending approval' and an 'Add Filter' button. At the bottom, there is a note: 'Filters allow you to select a subset of all the nodes to display. All Filters are ANDed together.'

12. Return to the **Add Filter** dropdown, and add the **Node: Published** filter. (As you add filters in Views, you may find that the interface hops back to the top of the page. If this happens, dutifully scroll back down to the **Filters** fieldset.) By default, when you add this filter, **Node: Published Equals Yes**. Leave the default as-is.

 If this filter is not added, the view will display items that are marked unpublished. This is an easy filter to forget, but it is recommended.

The screenshot shows the 'Filters' section with two filter configurations. The first filter is 'Node: Type' with 'Is One Of' operator and 'Swim Group' value. The second filter is 'Node: Published' with 'Equals' operator and 'Yes' value. Both filters have an 'Expose' button and a small blue icon in the Ops column. Below the filters, there is a text label 'Filter by whether or not the node is published. This is recommended for most Views!'.

13. You will now see a variety of **Ops** buttons. Note the **Delete**  button. With a little imagination, you can see it looks like a trash can. Also, if you move the mouse over the button, the popup text, **Delete this item** appears. The delete button does not delete content from the site, it simply removes the filter. For instance, if you delete the **Node: Published** filter, the view would contain both published and unpublished content. The up and down buttons will prove most useful in the **Sort** and **Field** fieldsets, which we cover in Chapter 2. In Chapter 4, we will also cover exposed filters.
14. Click on **Save**. When you first create a view, you are brought back to the "Views" **List** page. If you **Edit** your view from the view itself, you will be brought back to the view itself after editing it. Notice that the full URL of your view is `http://YOURSITE.com/swim-groups`. The message, **View successfully saved**, appears only the first time you view this. As an administrator, you will also see the **Edit**, **Clone**, and **Export** tabs. Those tabs are not displayed to site users who do not have Views administrative rights.

Swim Groups
View Edit Clone Export

View successfully saved.

Water Wrigglers (Ages 2 to 3)

Sun, 08/17/2008 - 15:52 — admin

For those new to the water. Basic skills such as floating, kicking, and blowing bubbles. The group is led by professional child educators who are also certified water instructors

Tadpoles (Ages 3 to 5)

Sun, 08/17/2008 - 16:27 — admin

Basic water fundamentals, including floating, kicking, paddling, and breath control.

Guppies (Ages 6 to 12)

Sun, 08/17/2008 - 16:28 — admin

Front and back float and paddle stroke. Rotary breathing, and front crawl techniques will be added to the classes as the student progresses.

Flying Fish (Ages 9 - 14)

Sun, 08/17/2008 - 16:28 — admin

Recipe notes

It is worth highlighting some subtle distinctions in naming conventions.

- **Name:** This is the unique identifier for a view. Hyphens are not allowed: use an underscore to separate multiple words.
Compare the Name fields for adding content and adding a view. When creating a content type in Drupal 5 (Step 2, above), the first item on the page is the *human-readable* **Name**, while the *machine-readable* type appears second. The Views interface (Step 8) breaks this convention; the first item on the page is the machine-readable **Name**.

Example:

Machine-readable: **swim_groups**

Human-readable: **Swim Groups**

It is not a big problem if you misname a view (by including spaces, for instance). All of your careful settings will remain intact, and Drupal will simply issue the message: **View name must be alphanumeric or underscores only**, when you attempt to save your view. Edit the **Name**, and scroll back down to the bottom of the page to **Save** the Views form again. You are not required to use lower case in your Views **Name**, but it is a common and oft-preferred practice.

It is acceptable for a machine-readable *view* name to be the same as the machine-readable *content type*. More typically, however, the content type will be singular, and the Views name plural. In our case, the content type is **swim_group**, and the view name is **swim_groups**.

- **URL:** If you would like to use more than one word in a URL, use hyphens to separate them. This enables Google (and possibly to other search engines as well) to index them as separate words. Many developers use underscores, but hyphens have the edge when it comes to search engine optimization.

Example:

View Name: **swim_groups**

View URL: **swim-groups**

You may include slashes in the URL field. For instance: **groups/swim**.

- **Title:** The title displays at the top of each page in the view. You may use some HTML markup such as `<em>`, `<i>`, and `<b>` to highlight phrases in the title. You may use a line break in a title `<br />`. You may also use HTML entities, such as `→`. A helpful list of entity codes can be found at: http://www.w3schools.com/tags/ref_entities.asp. For the most part, however, formatting of the full title is handled in your theme (thus, you will not specify fonts or other styles on the Views page).

Examples:

`<em>Swim</em> Groups`

Swim Groups

Swim Groups: `<br />`Tadpoles, and Guppies, and Wrigglers, Oh My!

Swim Groups:
Tadpoles, and Guppies, and Wrigglers, Oh My!

Voilà: Learn to swim at any age

Voilà: Learn to swim at any age

Compare *Views* Titles with *Node* Titles. HTML markup is allowed in View Titles, but not in Node Titles.

Recipe 4: Adding a Header to your View

[Ingredients
Completed Recipe 3]

You can easily add custom text or graphics above or below your view content, using the **Header** and **Footer** fieldsets. We will add a brief description at the top of the **Swim Groups** page.

1. Go to `http://YOURSITE.com/admin/build/views`. Your **swim_groups** view is now included in the list of views.
2. Click on **swim-groups** in the URL column.
3. Open the **Page** fieldset, and then open the **Header** fieldset contained within it. Enter the following header text:
Swimming is a great sport which exercises the whole body. Swim lessons are available for all ages and ability levels. Note: A parent or guardian must accompany children under 5 when in the water.
4. Open the **Input format** fieldset contained within the **Header** fieldset. Note that **Filtered HTML** is selected. See the *Recipe notes* if you would like to expand the list of available HTML tags.

5. Scroll down to the bottom of the **Edit view** page, and click on **Save**.
6. Enjoy the view.

Swim Groups

Swimming is a great sport which exercises the whole body. Swim lessons are available for all ages and ability levels. Note: A parent or guardian must accompany children under 5 when in the water.

Water Wrigglers (Ages 2 to 3)

Sun, 08/17/2008 - 15:52 — admin

For those new to the water. Basic skills such as floating, kicking, and blowing bubbles. The group is led by professional child educators who are also certified water instructors

Tadpoles (Ages 3 to 5)

Sun, 08/17/2008 - 16:27 — admin

Basic water fundamentals, including floating, kicking, paddling, and breath control.

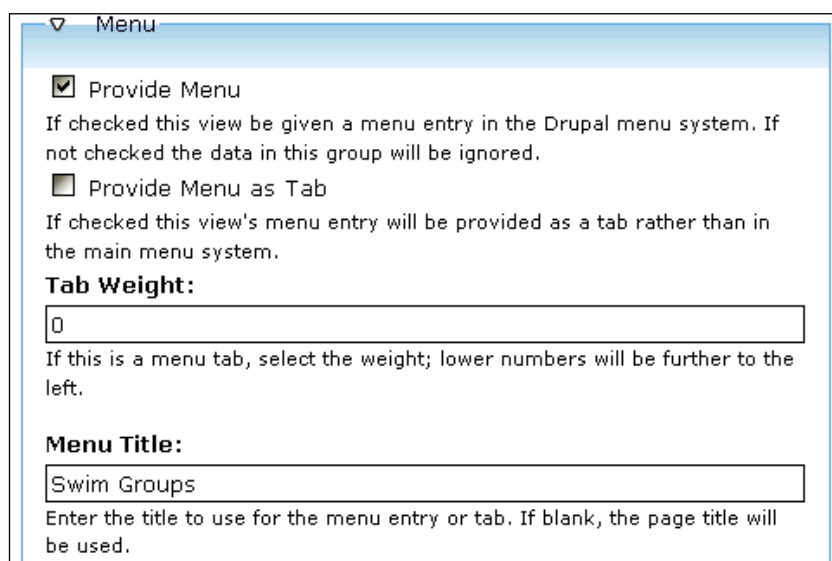
Recipe notes

- The **Header** fieldset (along with the **Footer** and **Empty Text** fieldsets) each contain **Input format** fieldsets. **Input format** filters will modify the display of text that has been entered by users (**Input format** filters are not connected to the filters fieldset that we described in *Recipe 3*). When a user enters text or code into a Drupal site, the text is saved in the database exactly as typed. Next time we view the text, however, it may not look exactly the same—some text and tags may not appear. To modify the list of allowed tags, visit the **Site Configuration | Input Formats** page, at: <http://YOURSITE.com/admin/settings/filters>.
- The most common reason for using these filters is to protect your site from malicious users, or even from users who don't intend to be malicious but who exhibit bad form. The **Input format** ensures that scripts will not be run, and that badly formed HTML will not break your site's layout. **Filtered HTML** is the name of one of the default **Input formats**.
- A terrific article by Robert Douglass offers helpful information on input filters at: http://www.lullabot.com/articles/drupal_input_formats_and_filters.

Recipe 5: Adding a View to a Menu

[Ingredients]
Completed Recipe 3 or 4

1. **Edit** the **Swim Groups** view, at: http://YOURSITE.com/admin/build/views/swim_groups/edit.
2. Open the **Menu** fieldset, found within the **Page** fieldset. Check the **Provide Menu** option, and add a **Menu Title**.



Menu

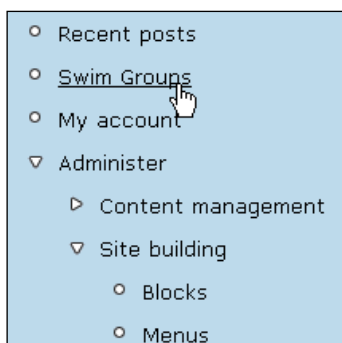
☒ Provide Menu
If checked this view be given a menu entry in the Drupal menu system. If not checked the data in this group will be ignored.

☐ Provide Menu as Tab
If checked this view's menu entry will be provided as a tab rather than in the main menu system.

Tab Weight:
0
If this is a menu tab, select the weight; lower numbers will be further to the left.

Menu Title:
Swim Groups
Enter the title to use for the menu entry or tab. If blank, the page title will be used.

3. By default, the menu item appears in the **Navigation** menu. We don't want it here so we will move it in the next step.



4. Go to **Administer | Site Building | Menus**.
(<http://YOURSITE.com/admin/build/menus>).
5. Use your browser to do a *Find* (usually *Ctrl+F*) for the word **Swim**. The first result will likely be in the actual sidebar menu item (if the Navigation menu has been placed in a region on the page). The second find will be the **Swim Group** content type listing in the menu configuration. The third one, finally, is the menu listing for the **Swim Groups** view. Click on the **Edit** link.
6. The **Path** is already filled in, based on our settings from the Views UI (Step 2). Enter a **Title**, and change the **Parent item** to **Primary Links**. If you wish the menu items to appear in alphabetical order, you may leave the **Weight** for all items at 0. In our example, we'll set the weight to a negative number, ensuring that it appears early among the primary links.

Edit menu item

Title: *

The name of the menu item.

Description:

The description displayed when hovering over a menu item.

Path:
swim-groups

☐ Expanded
If selected and this menu item has children, the menu will always appear expanded.

Parent item:

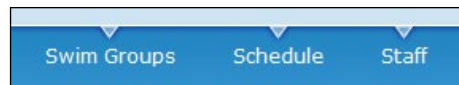
Primary links

Weight:
-2

Optional. In the menu, the heavier items will sink and the lighter items will be positioned nearer the top.

Submit

7. Your view is added to your **Primary Links** menu.



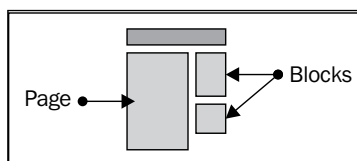
Recipe notes

Newly created swim groups are automatically added to the **Swim Groups** view, and its associated menu item.

Recipe 6: Creating a Block of Swim Groups

[Ingredients]
Completed Recipe 3 or 4

In *Recipe 3*, we created a page view, which is accessed via a URL. In this recipe, we are going to edit that view, adding a Block based on the same content. A view may provide both a **Block** and a **Page** view at the same time.



Blocks are placed inside regions made available by your theme. Sample regions that are common across most themes include the left sidebar, right sidebar, content top, and content bottom. A given block can appear on every page of the site, or just on pages that you designate. In the following steps you will create a **Block** for the **Swim Groups** view.

1. Go to **Administer | Site Building | Views**
<http://YOURSITE.com/admin/build/views>.
2. Select the **Edit** link next to the **swim_groups** view.
3. Open the **Block** fieldset, if it is not already open, and enter the following:
 - ☒ **Provide Block**
 - **View Type: List View**
 - **Title: Swim Groups**

- **Nodes per block: 10**
- ☒ **[More] Link?**

Block

☒ **Provide Block**
If checked this view will be provided as a block. If checked title may not be blank.

View Type:
List View

How the nodes should be displayed to the user.

Title:
Swim Groups
The title that will be shown at the top of the block. May be blank.

Nodes per Block:
10
If using a block, the maximum number of items to display in the block. Paggers are not used in blocks.

☒ **[More] Link?**
If using a view as both a page and a block, display a more link in the block that links to the view URL?

4. Open the **Fields** fieldset, and add the **Node:Title** field. You may leave the various options at their default settings.

Fields

Name	Label	Handler	Option	Sortable	Default Sort	Ops
Node: Title		Normal	As link	No	None	

Display the title of the node.

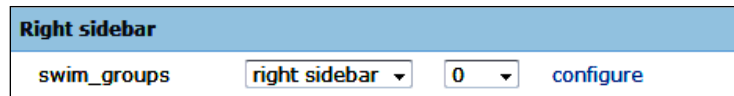
Add Field

Node: Title

Fields are only meaningful with List view and Table View; they allow you to choose which fields are presented and in what order.

5. Go to the **Blocks** page via **Administer | Site Building | Blocks**:
<http://YOURSITE.com/admin/build/block>.

Place the **swim_groups** block in the **right sidebar** (see the image below). Click on the **Save blocks** button at the bottom of the page.



- View your **Block** on any page of the site. The **Block** configuration page itself provides special formatting to inform administrators about various regions, so you may wish to view your block from a different page. For instance, click on the site logo, to go to the home page.



Recipe notes

It can sometimes be a bit disconcerting to have a single view create both a block and a page. The appearance may be different enough to be surprising. For instance, in our case, the block view contains a simple list of linked titles, while the page view contains the full text. The block had a maximum of **10** per page, while the page had a maximum of **20**. You may optionally create separate views for blocks and pages. Consider naming the respective views **block_swim_group**, and **page_swim_group**, for instance. (Use separate views when filters, fields, or sort order differs between the page view and the block view. Otherwise, it's more maintainable to use one view for both blocks and pages.) Note that the **Administer Views** list keeps track for you, signaling that the **swim_groups** view has provided a **Page**, a **Block**, and a **Menu** item.

Existing Views					
View	Title	Description	Provides	URL	Actions
swim_groups	New	List of swim	Page,	swim-groups	Edit
	Menu	groups	Block, Menu		Export Delete Clone

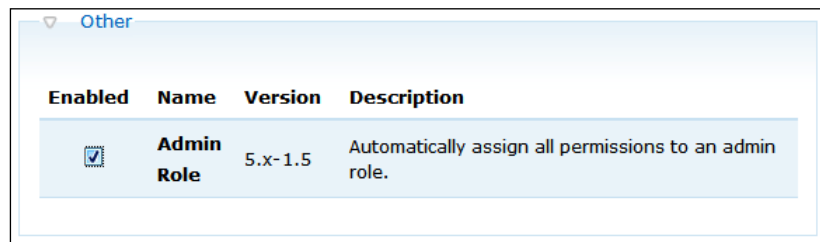
Recipe 7: Creating an "Admin" Role and setting Access Permissions

[**Ingredients**]
Admin Role module: <http://drupal.org/project/adminrole>

When you first installed Drupal, the first user you created was, naturally, user #1, also known as the "super-administrator". If others users need administrative access to the site (to administer Views, for instance), it is a good practice to create a distinct admin role. The **Admin Role** module will save you the trouble of having to manually set permissions each time you install a new module. All new permissions will automatically be enabled for the admin role.

Installing the Admin Role module

1. Go to <http://drupal.org/project/adminrole>.
2. Download the Drupal 5 **Admin Role** module to:
`<DRUPALROOT>/sites/all/modules`
(Select the "tarball" marked **Recommended for Drupal 5**.)
3. Unzip the module.
4. In your browser, go to the **Modules** page at **Administer | Site Building | Modules** (<http://YOURSITE.com/admin/build/modules>). Scroll down to the **Other** fieldset, and enable the **Admin Role** module.



Enabled	Name	Version	Description
☒	Admin Role	5.x-1.5	Automatically assign all permissions to an admin role.

Adding an Admin Role

- Go to the **Roles** page at **Administer | User Management | Roles** (<http://YOURSITE.com/admin/user/roles>) and add an **admin** role.

Roles

Roles allow you to fine tune the security and administration of Drupal. A role defines a group of users that have certain privileges as defined in [user permissions](#). Examples of roles include: anonymous user, authenticated user, moderator, administrator and so on. In this area you will define the *role names* of the various roles. To delete a role choose "edit".

By default, Drupal comes with two user roles:

- Anonymous user: this role is used for users that don't have a user account or that are not authenticated.
- Authenticated user: this role is automatically granted to all logged in users.

Name	Operations
anonymous user	locked edit permissions
authenticated user	locked edit permissions
admin	Add role

Configuring the Admin Role module

- Go to the **User management | Admin Role** page at (<http://YOURSITE.com/admin/user/adminrole>) and select your newly created **admin** role from the drop-down list.

Home > Administer > User management

Admin Role

Change which role is a "admin" Role with full perms

Admin Roles:

Which Role is Admin?

- Go to the **User Management | Access Control** page, and note that the **admin** (first column) already has the permissions set. These permissions were enabled by the **Admin Role** module.

Creating a User for the Admin Role

8. Go to **Administer | User Management | Users** and select **Add user**.
9. On the **User account** page for your new user, add a **Username**, **E-mail address**, and **Password**, and be sure to check the **admin** role.

Roles:
☒ admin
The user receives the combined permissions of the *authenticated user* role, and all roles selected here.

Recipe notes

You will receive an **Admin Permissions Set** message at the top of the module page each time you enable a new module (some modules do not actually create new permissions, but you will receive this status message, nevertheless).

Recipe 8: Installing Firefox and Firebug

Ingredients
Firefox: <http://www.mozilla.com/firefox>
Firebug: <http://getfirebug.com/>

The rest of the recipes in this chapter are devoted to enabling you to see the full scope of the **Views** interface. This recipe is a prerequisite for *Recipe 9, Revealing the full extent of Views*. Firefox and Firebug are also useful in any web developer's toolkit.

Installing Firefox

Bring up your current browser, and go to <http://www.mozilla.com/firefox>. Click on the **Free Download** button. Follow the detailed directions for installing Firefox here:

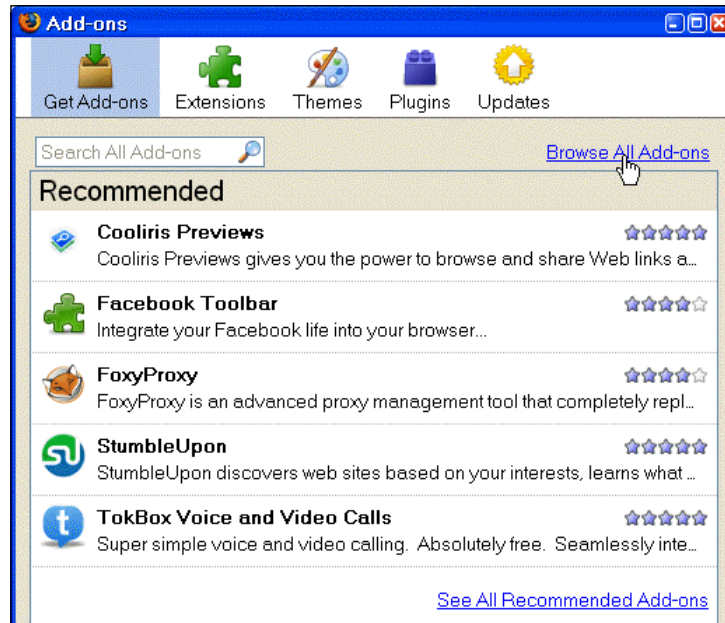
<http://support.mozilla.com/en-US/kb/Installing+Firefox>.

Installing Firebug

1. From Firefox, you can install Firebug by visiting **Tools | Add-ons**.

You may also download the Firebug add-on from: <https://addons.mozilla.org/en-US/firefox/addon/1843>, or <http://getfirebug.com/>

2. Go to the **Get Add-ons** tab, and click on **Browse All Add-ons**.

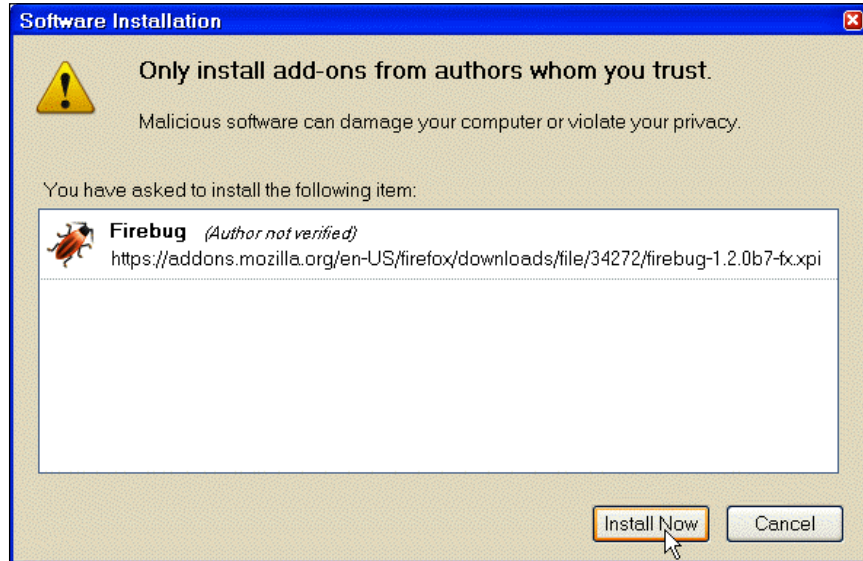


This will bring you to the **Firefox Add-ons** page.

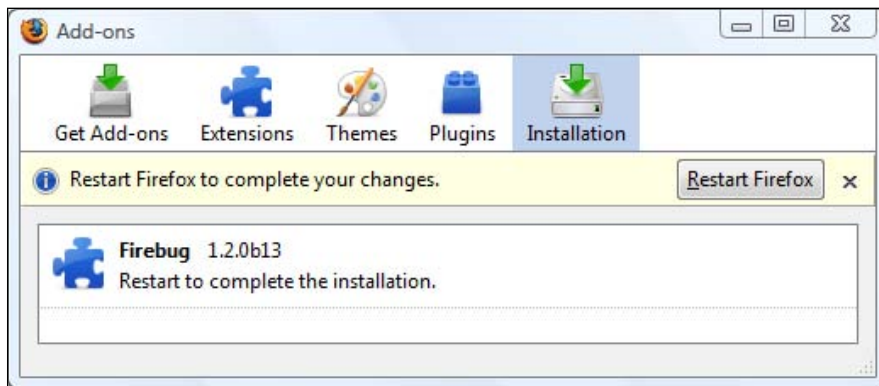
3. Enter **firebug** in the search box. When the add-on description is returned, click on **Add to Firefox**.



- Click on **Install Now**.



- Restart Firefox to make the Firebug add-on available to the browser.



- Note the new Firebug icon at the bottom-right corner of your browser. Congratulations! You now have Firebug installed.



Recipe notes

Firefox **Add-ons** are also called "extensions". Many web developers say that if they could only install one Firefox extension, it would be Firebug. Consider spending some time at getfirebug.com to learn more about its features. For more in-depth discussion, join the Firebug discussion group at <http://groups.google.com/group/firebug>.

Recipe 9: Revealing the full extent of Views

[Ingredients
Completed Recipe 8]

One reason that Views can feel a little bit mysterious, is that so much of it is concealed at first.

This recipe exposes all of the options on the main View's **Edit** screen by expanding all of the collapsed fieldsets on the page. This will save you the time it would take to manually open the nearly two-dozen fieldsets. The recipe uses a single line of jQuery. Please do not feel that you have to understand everything all at once when you see the full extent of Views! That's what the rest of this book is for. There is value, however, in the full array of options that will be available to you. A summary of the full screen can be found in *Recipe 7*. Note that fieldsets are nested within other fieldsets, up to three levels deep.

The list of all the fieldsets in the View's **Edit** screen is as follows:

- Basic Information
- Page
 - Header
Input format
 - Footer
Input Format
 - Empty Text
Input format
 - Menu
Default Menu Tab
- Block
 - Header
Input format
 - Footer
Input format
 - Empty Text
Input format
- Fields
- Arguments
 - Argument Handling Code
- Filters
- Exposed Filters
- Sort Criteria

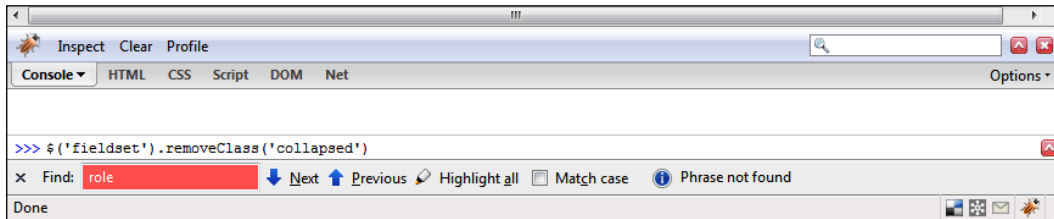
Ensure that Firefox and Firebug are installed (See Recipe 8)

1. Go to `admin/build/views/swim_groups/edit`. If you have not yet created a view, go to `admin/build/views`, and click on **Add**.
2. Click on the firebug icon  in the Firefox status bar, or press *F12* (on Apple laptops, it may be necessary to hold down "Fn" while pressing *F12*). If you get a message regarding the need to enable Firebug for the site, go ahead and do that. Firebug opens at the bottom of the browser.
3. Click on the **Console** tab in Firebug. An understated JavaScript (and jQuery) command line appears in the lower left, after the `>>>` prompt. You may notice the blinking cursor.

Running the jQuery command

4. In the command line area, enter the following jQuery command. Make sure to include the dollar sign character.

```
$('fieldset').removeClass('collapsed')
```



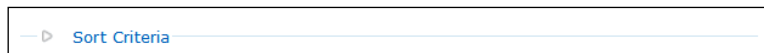
5. The full scope of the views **Add** or **Edit** interface appears in the browser. Take some time to explore the page.

Recipe notes

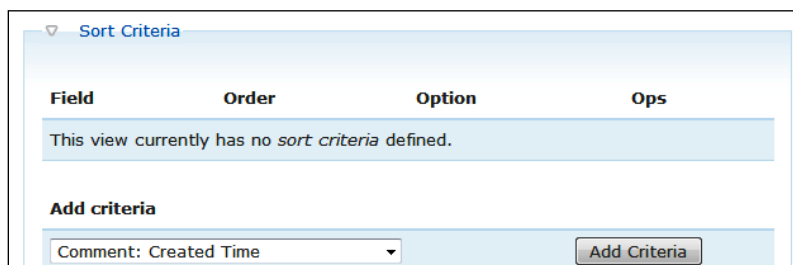
How does this recipe work? If the jQuery command above were to be read as English, it would say: "Find all of the fieldsets on the page, and remove the collapsed CSS class from all of them". The effect is to open all the fieldsets (JavaScript must be enabled for this to work).

Let's elucidate this further:

- The HTML fieldset tag in most Drupal themes (including the default *Garland* theme) looks like this, when collapsed:
- `<fieldset class="collapsible collapsed">`



- An expanded tag looks like this in HTML:
`<fieldset class="collapsible">`



The jQuery code removes all of the "collapsed" classes from the Views page. Thus, all the fieldsets are open, revealing their full contents.

Most of the time, you will not want to interact with Views in this expanded way, but it certainly is helpful, sometimes. It's nice to know that Firebug CSS edits are temporary. The next time you view the page, the fieldsets will refresh with their default open or closed appearance. Press *F5* in Firefox to refresh the page you are on.



This jQuery fieldset expansion command does not function if you have already manually closed a fieldset with your mouse.

Recipe 10: Preparing Views Worksheets

Ingredients

The Views Worksheet in this chapter
Copy machine
Pen or highlight markers

On the following page is worksheet that lists each element on the View's **Add** or **Edit** form. This worksheet will serve as your guide as you learn the features of Views.

1. You have permission to photocopy the **Views Worksheet**. Start with a dozen copies or so. You may choose to copy more or less depending on whether you think this resource suits your learning style.
2. Edit the **Swim Groups** view that you created in *Recipe 3*
http://YOURSITE.com/admin/build/views/swim_groups/edit.
3. On a worksheet copy, mark each element used by the view (use a pen, or highlight markers according to your preference).

Recipe notes

The worksheet divides elements into four categories: **Text**, **Checkboxes**, **Dropdowns**, and **Radio Buttons**.

- Some text boxes expect numeric characters, as noted in parentheses.
- Checkboxes can be considered as a yes or no question. For instance: **Provide Page View** is the Views shorthand for "Do you want to provide a page view?"
- Some of the drop-down boxes allow you to select more than one element.
- Radio buttons are mutually exclusive.

Views Worksheet Make copies to track your learning progress. View Name: _____			
Text	Checkboxes	Drop-downs	Radio Buttons
Basic Information			
Name	Access Roles		
Description			
Page			
URL	Provide Page View	View Type	Header: Inputformat
Title	Use Pager		Footer: Inputformat
Nodes per Page (Numeric)	Breadcrumb trail should not include "Home"		Empty Text Inputformat
Header			
Footer			
Empty Text			
Menu (This is a fields et nested inside the Page fields et)			
Tab Weight (Numeric)	Provide Menu	Default Menu Tab:	
Menu Title	Provide Menu as Tab	Parent Menu Item Type	
Default Menu Tab:	Default Menu Tab:		
Tab Weight (Numeric)	Make Default Menu Tab		
Default Menu Tab:			
Parent Menu Item Title			
Block			
Title	Provide Block	View Type	Header: Inputformat
Nodes per Block (Numeric)	[More] Link?		Footer: Inputformat
Header	Header: Use Page Header		Empty Text Inputformat
Footer	Footer: Use Page Footer		
Empty Text	Empty Text: Use Page Empty		
Fields			
Label		Handler	
		Option	
		Sortable	
		Default Sort	
		Add Field	
Arguments			
Title		Default	
Option (may also be drop-down)		Option (may also be text)	
Wildcard		Add Argument	
Wildcard Sub			
Argument Handling Code :			
Argument Code			
Filters			
Value (may also be dropdown)		Operator	
Option		Value (may also be text)	
		Add Filter	
Exposed Filters			
Label	Optional		
	Filter settings Default		
	Force Single		
	Lock Operator		
Sort Criteria			
Option (may also be drop-down)		Order	
		Option (may also be text)	
		Add Criteria	

Summary

In Chapter 1, we installed Views, took a whirlwind tour of associated URLs, and then created our first view: a list of **Swim Groups**. We edited the page view to add header text and then created a menu link. We then produced a block that could be added to any page. We had a chance to see the full scope of the Views interface in two ways: first, using a jQuery command to expand all fieldsets, and second, using a summary worksheet of all the Views Edit options. Worksheet copies will continue to be a useful resource throughout the rest of the book.

2

Working with Default Views

Right out of the box, the Drupal **Views** module provides six pre-made "default" views that you can rapidly incorporate into your site. While serving your site, the default views can also serve as a valuable learning resource, providing keys to understanding a number of advanced Views features. Using default views as examples we will have the chance in this chapter to introduce all but one of the primary Views fieldsets.

With only core Drupal and **Views** installed, you will initially see just four default views. There are displayed at the bottom of the page at <http://YOURSITE.com/admin/build/views>.

Two more default views become available when you enable the core **Statistics** module.

Administer views						
List Add Import Tools						
This screen shows all of the views that are currently defined in your system. The default views are provided by Views and other modules and are automatically available. If a customized view of the same name exists, it will be used in place of a default view. [more help...]						
No views have currently been defined.						
Below are system default views; if you edit one of these, a view will be created that will override any system use of the view.						
Default Views						
Default view	Title	Description	Provides	URL	Status	Actions
comments_recent		Display recent comments block	Block	No Page View	Enabled	Add Disable
frontpage		The basic front page view.	Page	frontpage	Enabled	Add Disable
popular_alltime	All-time popular content	Nodes sorted by total page-views	Page, Block, Menu	popular/alltime	Disabled	Add Enable
popular_recent	Recent popular content	Nodes sorted by recent page-views	Page, Block, Menu	popular/latest	Disabled	Add Enable
taxonomy_term	Taxonomy	The taxonomy view with a depth of 0.	Page	taxonomy/term	Disabled	Add Enable
tracker	Recent posts	Shows all new activity on system.	Page, Menu	tracker	Enabled	Add Disable

Default views are created in code (by contrast, most views reside in the views tables in the Drupal database). Any contributed module may create one or more default views. *Appendix A* contains a list of close to 200 default views from sixty contributed Drupal 5 modules.

For the default views that come with Views module, the code is found in the following directory:

```
<DRUPALROOT>/sites/all/modules/views/modules
```

The following table provides information about the six default views packaged with the **Views** module. The table shows *which file created the view*, and *which fieldsets are utilized*. For comparison, we'll also show which fieldsets were implemented in the two views we created in Chapter 1.

	Basic Info	Page	Menu*	Block	Fields	Arguments	Filters	Exposed Filters	Sort
comments_recent									
views_comment.inc (Recipe 11)	✓			✓	✓		✓		✓
frontpage									
views_node.inc (Recipes 12 and 13)	✓	✓				✓	✓		✓
popular_alltime									
views_statistics.inc (Recipes 14 and 15)	✓	✓	✓	✓	✓		✓		
popular_recent									
views_statistics.inc (Recipes 14 and 15)	✓	✓	✓	✓	✓		✓		
taxonomy_term									
views_taxonomy.inc (Recipes 16, 17, and 18)	✓	✓				✓	✓		✓
tracker									
views_user.inc Recipes 19, 20, and 21	✓	✓	✓		✓	✓	✓		✓

	Basic Info	Page	Menu*	Block	Fields	Arguments	Filters	Exposed Filters	Sort
swim_groups (Recipes 3 and 4)	✓	✓					✓		
swim_groups (Recipe 6)	✓	✓		✓	✓		✓		

*The Menu fieldset is nested inside of the Page fieldset

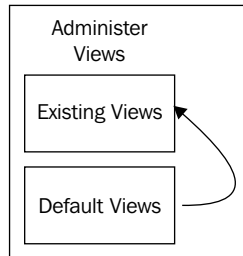
Recipe 11: Adding "Recent Comments" to your site

The **comments_recent** default view creates a block of recent comments. We will use the standard **Blocks** administration page — which by now is likely familiar to you — to place the block into a page region. Yet there are some prerequisites and quirks that are worth getting to know:

- The **comments_recent** default view is available only if the *comments module is enabled* (in a standard Drupal installation the comments module is enabled by default).
- Even if the comments module is enabled, a default block view appears in the Blocks Administration page only if the *default view is enabled*. (The **comments_recent** view is enabled by default. Click on **Enable** in the **Actions** column to change a **Disabled** default view **Status** to **Enabled**.)

Default Views						
Default view	Title	Description	Provides	URL	Status	Actions
comments_recent		Display recent comments block	Block	No Page View	Enabled	Add Disable
frontpage		The basic front page view.	Page	frontpage	Enabled	Add Disable
taxonomy_term	Taxonomy	The taxonomy view with a depth of 0.	Page	taxonomy/term	Disabled	Add Enable

- I recommend that you override default views before using them. Simply click on **Add** to start an override. You may modify view settings, or even leave the existing settings intact before scrolling to the bottom to click on **Save**. When you override a default view, the status changes to **Overridden** and a new view appears in the top of the `admin/build/views` page, under **Existing Views**. This view is stored in the database.



The block would actually show up perfectly well if you did not choose to override the default. However, in order to make any modifications to a default view through the Views UI interface, you will need to perform this override. Also, occasionally, some theming functions will only work properly with an overridden view.

- When you visit `admin/build/blocks` you will actually see *two* **Recent comments** blocks. This confuses many people, but the reason is simple: two different modules create separate blocks by the same name.

Recent comments block created by the Comment module	Recent comments block created by the Views module
Source Code: <code>modules/comment/comment.module</code>	Source Code: <code>sites/all/modules/views/modules/ views_comment.inc</code>



Armed with this information, we are now ready for our recipe:

Ingredients

Ensure that the prerequisites described above are met.

(The comments module is enabled, and the view is enabled.)

Overriding the View

1. Go to <http://YOURSITE.com/admin/build/views>. Override the **comments_recent** view by selecting **Add**. Scroll down to the bottom of http://YOURSITE.com/admin/build/views/add/comments_recent, and **Save** the view.

Configuring the Block

2. Go to <http://YOURSITE.com/admin/build/blocks>. Place your cursor over one of the **Recent comments-configure** links (depending on override status, the blocks may be labeled **Recent coments** or **recent_comments**).



Lower-case block names generally appear below capitalized ones, so look near the bottom of the list of blocks if you do not see two **Recent comments** blocks next to each other.

You can configure the behaviour of each block (for example, specifying on which pages and for what users it will appear) by clicking the "configure" link for each block. [\[more help...\]](#)

Block	Region	Weight	Operations
Left sidebar			
User login	left sidebar	0	configure
Navigation	left sidebar	0	configure
Disabled			
All-time popular content	<none>	0	configure
Primary links	<none>	0	configure
Recent comments	<none>	0	configure
Recent comments	<none>	-1	configure
Syndicate	<none>	0	configure
Who's new	<none>	0	configure
Who's online	<none>	0	configure

- With your mouse still hovering over **configure**, check the status bar in your browser. If it reads:

```
http://YOURSITE.com/admin/build/block/configure/views/
comments_recent
```

then you have identified the default view block. If it says:

```
http://YOURSITE.com/admin/build/block/configure/comment/0
```

you've got the view defined by the comment module.

The main reason to use the views-created block is that it enables quick and flexible modifications. If you are perfectly happy with the default block from this module, you may use it.

4. Place the block in the region of your choice. Common choices are **left sidebar**, **right sidebar**, and **content bottom**. Click on the **Save Blocks** button.
5. Scroll up the page to find the block in your new region's section. Click on **Configure** to change the title. If you enter nothing in the block title, the default block title will be used; the view uses a lower-case "c" in the title. If you prefer proper case (**Recent Comments**) you could enter that here (alternatively, you can change the title on the views edit page). As usual, you may also use the block configuration page to specify visibility by user, role, and page. Save the block.

Configuring the Date Field

6. Return to `admin/build/views`, and select **Edit**.
7. In Chapter 1, *Recipe 7*, we set permissions for the admin role to access all views. However, we did not provide these permissions for anonymous and authenticated users. Be sure to add access for these roles when you edit the view. Use the **Access** checkboxes in the **Basic Information** fieldset.
8. Visit the **Fields** fieldset, and change the **Handler** for **Comment: Created time**, from **As Short Date** to **As Time Ago**.

The screenshot shows the 'Comment' field configuration in a Drupal Views interface. The 'Handler' dropdown menu is open, displaying a list of options: 'As Time Ago' (which is highlighted and selected), 'As Short Date', 'As Medium Date', 'As Long Date', 'As Custom Date', and 'As Time Ago'. To the right of the dropdown, there is a 'No' checkbox that is checked, and a 'None' dropdown menu that is also open. The interface includes various icons for field configuration, such as a square, a double arrow, and an up arrow.

Recipe notes

- The guidance in Step 7 to enable appropriate roles applies to all the Views recipes throughout the book.
- We will review more Date formatting options in Chapters 4 and 7.
- A user prepared a **Comment: Distinct** filter patch, which you may find to be helpful. It can be found at: <http://drupal.org/node/219499>.



Recipe 12: The frontpage View

[Ingredients
Views RSS module]

The **frontpage** view serves two basic functions. It presents key content on the site's home page, and also enables the home page content to be syndicated with a **Really Simple Syndication (RSS)** feed.

The default **frontpage** view includes content marked for the front page, keeping items marked **Sticky** at the top. It is common to modify this default, particularly to filter for selected content types.

Give some thought to which content types you wish to include on your home page. In the **MySafeWork.com** screenshot, for instance, only **Blog** entries appear in the **frontpage** view, while **Videos** and other blocks are placed around the site's edges. Another site could set up Events and News to stream to the front page, keeping other content items, such as Forums and Slideshows, available only through the navigation menu.

The **frontpage** view introduces five views features. We'll highlight two (both noted with a magnifying glass) with some detail before beginning the recipe.

Page Fieldset

- **View Type: Teaser List** 🔍
- Breadcrumb trail should not include "Home"

Arguments

- **Node: Feed Selector** 🔍

Filters

- **Node: Front Page Equals Yes**

Sort:

- **Node:Sticky**



Teaser List

This is the first recipe in the book that uses the **Teaser List-View Type**. The teaser includes the first portion of the post, along with a **Read More** link to the full text. The Read More link typically appears in a separate line, together with **Comments**, and **New comments** links. *Recipe 13*, below this one, offers a nice tweak to make the **Read More** link easier to find. You may alternatively choose to show the **Full Nodes** or a **List View**. If either **Full Nodes** or **Teaser List** is selected, it is not necessary to populate the **Fields** fieldset.



Node Feed Selector Argument

Arguments are used to modify a view, and each argument offers a different type of modification. Both the frontpage and **taxonomy_term** default views use a special argument called **Node: Feed Selector**, to create an RSS Feed. There is a prerequisite to activating this argument – the **Views RSS** module must be enabled for the `/feed` parameter to work.

Now we will begin the recipe.

Enabling the Views RSS module

1. Go to <http://YOURSITE.com/admin/build/modules>, and enable the **Views RSS** module.

☒	Views RSS	5.x-1.6	RSS plugin for the views feed selector argument. Depends on: Views (enabled)
-------------------------------------	------------------	---------	--

Looking at the Frontpage View, and Overriding it

2. Go to <http://YOURSITE.com/admin/build/views>. Click on the frontpage URL. If your site has content designated for the frontpage, you will see up to ten items in the view. You should also see (depending on your theme) a feed icon at the bottom of the page.



3. Once you feel familiar with how the view looks, click the **Override** tab.

Basic Information fieldset

4. In most cases the view description is only for the benefit of the view administrator. Since this view has a public feed, you may wish to change **The basic front page view** to something far more reflective of your specific content. For now, we will leave this intact, choosing instead to override this description via an argument option, below.

Page fieldset

5. Review the **Page** fieldset. Note the **Teaser List**, discussed above.
6. **Breadcrumb trail should not include "Home"** is specially designed for a frontpage view. Breadcrumbs are the links that track the current page's location in a site hierarchy, for example, **Home | Activities | Swim Groups**. If you're already on the home page, you will not likely want a home link in the breadcrumb.

Fields fieldset

7. Notice that there are no fields selected, since we are using a **Teaser List**.

Feed Selector Argument

8. Scroll down to the **Arguments** fieldset. Note the following settings:

Argument Type	Default	Title	Option	Wildcard	Wildcard Sub	Ops
Node:						
Feed	Display All Values	Camp Teval				
Selector						

- **Default**

It's certainly not obvious, but you will want to leave the **Default** argument handler as **Display All Values** for the **Node: Feed Selector** argument. Other settings in this particular handler dropdown will yield an unsightly SQL error message. There is some brief discussion about this behavior in the Views issue queue, at: <http://drupal.org/node/136200>.

- **Title**

The feed title for the **frontpage** view is pre-filled in with the site name, followed by a hyphen, followed by the site slogan (if there is one). The underlying code for this is found in the `node_views_default_views()` function in `views_node.inc`.

```
'title' => variable_get('site_name', 'drupal') . ' - ' .  
variable_get('site_slogan', ''),
```

The title is getting the site name from the `variable` table. If it does not find one, it fills in `drupal` as the title. If there is no site slogan, there will be an extraneous hyphen. (Scroll to the right in the title box to see the hyphen. Feel free to go ahead and delete it, if you do not want to use a slogan.)



Clear Views Cache

If you have recently changed the site name or slogan, and it does not appear here, you would likely need to clear your views cache before overriding the default view. The clear views cache button is found in the tools tab, at <http://YOURSITE.com/admin/build/views/tools>.

- **Option**

The purpose of the option varies by specific argument. For the Feed Selector argument, the option sets the subtitle of the feed, overriding the text of the view description. The `node_views_default_views()` function offers this preset:

```
'options' => variable_get('site_mission', ''),
```

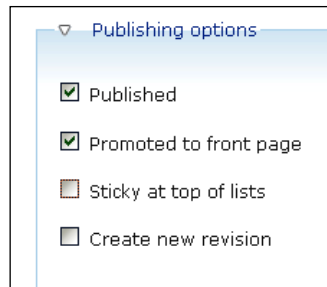
In other words, if you have a site Mission, it will become the default subtitle of the feed. The site Mission is set at <http://YOURSITE.com/admin/settings/site-information>. If the site Mission exists, the theme does not need to have the Mission enabled for this Views prefill to work. The site's Mission statement is not required as a feed title. Type any subtitle here. Custom **Option** text will appear only in the feed: not on the front page itself.

- **Wildcard and Wildcard Substitution**

Leave these two items blank for the **frontpage** view (most views leave this blank).

Filters fieldset

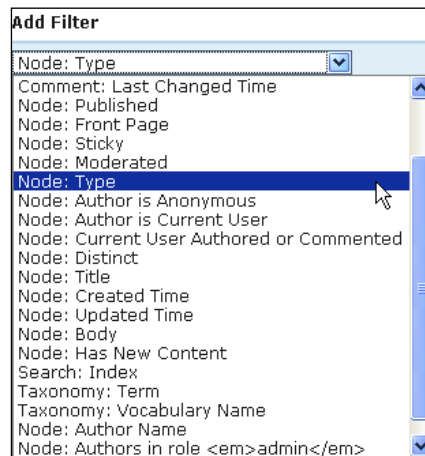
9. Scroll down to the **Filters** fieldset. The default filters are **Node: Front Page Equals Yes**, and **Node: Published Equals Yes**. In this case, **Front Page** is not a reference to the **frontpage** view, itself. Rather, it refers to items that are promoted to front page, for which the `promote` field in the node table is set to **1**.



Publishing options

- ☒ Published
- ☒ Promoted to front page
- ☐ Sticky at top of lists
- ☐ Create new revision

Typically, you will want to add an additional filter for **Node: Type**. Note that the filters are not in alphabetical order. Node fields appear in the order designated by the `node_views_tables()` function, in `views_node.inc`.



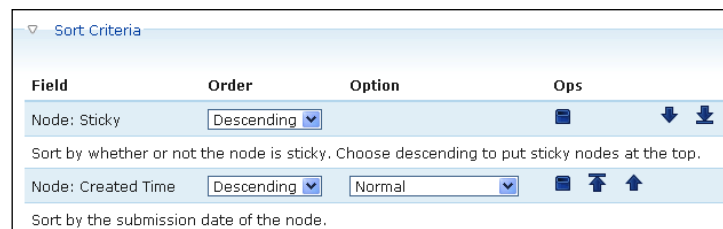
Add Filter

Node: Type

- Comment: Last Changed Time
- Node: Published
- Node: Front Page
- Node: Sticky
- Node: Moderated
- Node: Type**
- Node: Author is Anonymous
- Node: Author is Current User
- Node: Current User Authored or Commented
- Node: Distinct
- Node: Title
- Node: Created Time
- Node: Updated Time
- Node: Body
- Node: Has New Content
- Search: Index
- Taxonomy: Term
- Taxonomy: Vocabulary Name
- Node: Author Name
- Node: Authors in role admin

Sort Criteria fieldset

- Review the **Sort Criteria** fieldset. The **frontpage** view places "Sticky" nodes at the top, and then performs a secondary sort by **Created Time**. Typically, Drupal themes will create a `<div>` with a sticky class so that these important items can be styled differently via CSS.



Sort Criteria

Field	Order	Option	Ops
Node: Sticky	Descending		  
Sort by whether or not the node is sticky. Choose descending to put sticky nodes at the top.			
Node: Created Time	Descending	Normal	  
Sort by the submission date of the node.			

Note the helpful text, such as **Choose descending to put sticky nodes at the top**. This is informative help text was created by the `node_views_tables()` function in `views_node.inc`.

11. **Save** the View.

Enabling the frontpage View to show on the front page

12. Many Drupal users stop before this step and then wonder why their well-crafted view does not actually show up on the home page. There are several ways to ensure that the view lands on the front page. A quick way is to go to:

Administer | Site Configuration | Site Information

<http://YOURSITE.com/admin/settings/site-information> and change the **frontpage URL** from **node** to **frontpage** (the name of your view).



Default front page:
`http://localhost/d5/?q=`
The home page displays content from this relative URL. If unsure, specify "node".

A second approach is to use the **Panels** module, which we will look at in Chapter 7.

13. Go to the **frontpage** view and look for the orange feed icon. In selected browsers, such as Firefox, you may see a feed icon in the browser address bar. If you have a different browser, scroll until you see the feed icon (in most themes, this is located at the bottom of the page).
14. This is how the feed looks in Firefox. You will also see a URL which you can import into the feed reader of your choice (see the *Recipe notes* for synonyms for "feed reader").



Removing duplicate search engine results

It is good practice to prevent search engines from indexing duplicate pages. For instance, after moving the front page to the actual home page, there is no need for search engines to index the separate `http://YOURSITE.com/frontpage` URL.

15. Log on to your host, and edit the `robots.txt` file in your Drupal home directory. Add the following line to the `# Directories` section:

```
Disallow: /frontpage
```

(From the point of view of a search engine, this URL involves a frontpage "directory" — even though there is no such directory on your system.)

Recipe notes

- Applications that read your feed URL are known by a variety of names, including: Feed Reader, RSS Reader, News Aggregator, Feed Aggregator, and RSS News Reader. Some popular such readers include Google Reader, Bloglines, MS Outlook, and Firefox.
- You may be used to thinking that people are accessing your content via the website. However, people are increasingly viewing web content via these applications. For the fun of it, try Googling "The basic front page view" to see a number of sites which neglected to customize the default description.

- There are actually two ways to override a default page view from the **Administer views** page. In *Recipe 11*, we selected **Add**. In this recipe we chose to click on the view's URL, and then click the **Override** tab (the URL method was not available for the **comments_recent** block). These two approaches yield two different URLs:
`http://YOURSITE.com/frontpage/add`
`http://YOURSITE.com/admin/build/views/add/frontpage`
However, they both serve to successfully override the default view, as long as you remember to click on the **Save** button.
- If you care about the precise order of items on your home page, consider using the **Nodequeue** module:
`http://www.drupal.org/project/nodequeue`
- Information on `robots.txt` is found at:
`http://tips.webdesign10.com/robots-txt-and-drupal`
`http://www.robotstxt.org/wc/robots.html`
- We will share a technique for theming the frontpage view in the notes for *Recipe 79*.

Recipe 13: Read More link

Ingredients

Read More link module: `http://drupal.org/project/ed_readmore`

The **Read More link** module moves the standard **Read more** link into the teaser body, where it is generally easier for users to notice. The module applies universally to teasers, whether content is part of a view or not, but seems especially appropriate to recommend it together with the **frontpage** view (*Recipe 12*).

1. Note the appearance of the **Read more** link in your content item.



2. Install and enable the **Read More link** module.
3. Note the new location of your read more link. You may configure the text of the link at http://YOURSITE.com/admin/settings/ed_readmore.



Recipe notes

- This tweak has the added benefit of adding a link to the RSS feed.
- Add a `<!--break-->` comment in the body of your content to control the teaser text break, or set a default teaser length at **Administer | Content Management | Post Settings**.

<http://example.com/admin/content/node-settings>

- The default teaser length is not retroactive to previously entered content.
- Consider using the **Node Teaser** or **Excerpt** modules if you want your teaser text to be different than the beginning of the article.

<http://drupal.org/project/nodeteaser>

<http://drupal.org/project/excerpt>

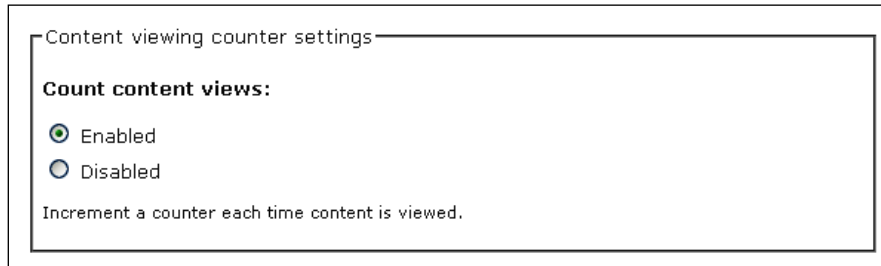
Recipe 14: All-time Popular, and Recent Popular Pages

[Ingredients]
Statistics module, enabled

Site visitors are generally curious about the content other visitors enjoy. The **popular_alltime** and **popular_recent** default views are associated with the **Statistics** module. Both views create a block and a tabbed page. In this recipe we look at the tabbed pages.

Enabling Statistics

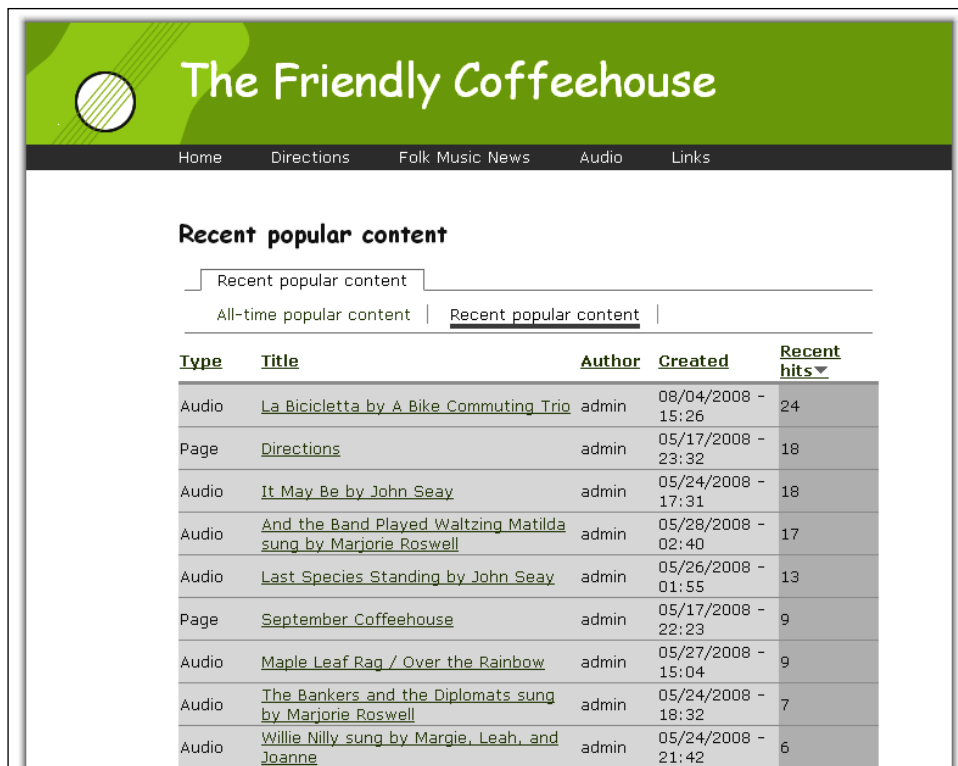
1. Enable the core **Statistics** module at `/admin/build/modules`, if it is not already enabled.
2. Go to **Administer | Logs | Access log settings**, at `admin/logs/settings`. Enable **Count content views**.



If you do not take this step, the two statistics views will not know how to determine what's popular, and the **Reads** or **Recent Hits** columns will be empty. This step is not retroactive, so if your site has been running for a year, but you've only just enabled the content view counts, your counters will start at 0.

Overriding the Default Views

3. Go to the **Administer views** page, and enable both the **popular_recent**, and **popular_alltime** views (the **Views** module disables these views by default; the **Enable** button is found in the **Actions** column on the `admin/build/views` page).
4. Go to `http://YOURSITE.com/popular`. If you have not yet overridden your two default views, you are likely to see a **Page Not Found** error message.
5. If you receive this message, fear not. Simply go back to `admin/build/view`, and override the two "popular" default views. You may use either the **Add or URL | Override** method to do this. The **Add** method requires just two clicks, so the quickest sequence of clicks is: **Add** | scroll down | **Save** for both views.
6. Visit `http://YOURSITE.com/popular` again. You will now see the two views, complete with tabbed navigation. In other words, the ability to successfully use these views depends on them being overridden (technically, just the **popular_recent** view needs to be overridden, but it may be easier to remember just to override them both).



The screenshot shows the website 'The Friendly Coffeehouse' with a green header and a navigation bar. The 'Recent popular content' tab is selected and underlined. Below the tabs is a table of content items.

Type	Title	Author	Created	Recent hits▼
Audio	La Bicicletta by A Bike Commuting Trio	admin	08/04/2008 - 15:26	24
Page	Directions	admin	05/17/2008 - 23:32	18
Audio	It May Be by John Seay	admin	05/24/2008 - 17:31	18
Audio	And the Band Played Waltzing Matilda sung by Marjorie Roswell	admin	05/28/2008 - 02:40	17
Audio	Last Species Standing by John Seay	admin	05/26/2008 - 01:55	13
Page	September Coffeehouse	admin	05/17/2008 - 22:23	9
Audio	Maple Leaf Rag / Over the Rainbow	admin	05/27/2008 - 15:04	9
Audio	The Bankers and the Diplomats sung by Marjorie Roswell	admin	05/24/2008 - 18:32	7
Audio	Willie Nilly sung by Margie, Leah, and Joanne	admin	05/24/2008 - 21:42	6

7. Review the following information on menu options.

The Menu Options

Tabbed navigation enables each view to have a link to the other specified views. Thus, the Views module offers a powerful and easy-to-implement navigation system. In the theme displayed above, the current tab, **Recent popular content**, is underlined. It is also designated in the view as the default menu tab. That's why it is available at <http://www.friendlycoffeehouse.org/popular>, in addition to its officially-named URL, <http://www.friendlycoffeehouse.org/popular/latest>.

Tabs are Menus?



In Drupal code, navigation tabs are part of the menu system. These tabs are called "menu local tasks". One default tab may use the parent path. Views menu options designated as tabs will not appear in the Drupal menu settings at `admin/build/menu`. See *Recipe 5* to add a view to a standard web page menu.

Here is a summary table, comparing the menu functions used by the views. Filled-in circles indicate that the option is used.

Menu Options in the <code>popular_recent</code> and <code>popular_alltime</code> default views		
Fieldset Option	All-time	Recent
Page Menu Provide Menu	●	●
Page Menu Provide Menu as Tab	●	●
Page Menu Tab Weight	●	●
Page Menu Menu Title (<i>defaults to Page Title if blank</i>)	○	○
Page Menu Default Menu Tab Make Default Menu Tab	○	●
Page Menu Default Menu Tab Parent Menu Item Type	○	●
Page Menu Default Menu Tab Tab Weight	○	○
Page Menu Default Menu Tab Parent Menu Item Title (<i>defaults to menu title, or Page Title, if blank</i>)	○	○

- Go to `admin/build/views`, and **Edit** the `popular_recent` view. Remember to look under **Existing Views**, and not the **Default Views**.
- Open the **Page** fieldset, and note the URL. The `popular/latest` name signals that it is in the same tab set as the `popular/all` (**popular_alltime**) view. If you created another view and gave it the URL `popular/music` (even if it had nothing to do with the **Statistics** module), it would appear with tab links to the other two views.
- Open the **Menu** fieldset within the **Page** fieldset. Both `popular_recent` and `popular_alltime` provide a menu, and enable it as a tab. If we wanted to have **Recent Popular Content** tab link appear to the left of **All-time Popular content**, we could give it a negative tab-weight.

Menu

☒ Provide Menu
If checked this view be given a menu entry in the Drupal menu system. If not checked the data in this group will be ignored.

☒ Provide Menu as Tab
If checked this view's menu entry will be provided as a tab rather than in the main menu system.

Tab Weight:

If this is a menu tab, select the weight; lower numbers will be further to the left.

Menu Title:

Enter the title to use for the menu entry or tab. If blank, the page title will be used.

—▷ Default Menu Tab

11. Expand the **Default Menu Tab** fieldset. The first checked item enables **popular_recent** to be the default tab.

Default Menu Tab

☒ Make Default Menu Tab

If checked this view's menu entry will be provided as a tab, and will be the default tab for that URL path. For example, if the URL is 'tracker/all' and it is set as the default menu tab, it will be put into the menu as 'tracker' and 'tracker/all' will be the default tab. The following settings allow you to customize the parent item, for example 'tracker'. For tabs to work properly, one tab in the group must be set as the default.

Parent Menu Item Type:

Tab

Select type of parent item to use for this default menu tab. You can either specify the parent should be a tab (the default), a normal menu item, or to use the menu item that already exists at the specified URL. For example, if the URL for the default tab is 'tracker/all', then 'tracker' would already have to be a valid menu item to use this final choice.

Tab Weight:

0

If the parent menu item is a tab, select the weight; lower numbers will be further to the left.

Parent Menu Item Title:

If the Parent Menu Item is being defined by this view (if you set the *Parent Menu Item Type* to either *Tab* or *Normal menu item*), you can specify its title here. If blank, the menu title will be used if that is defined, or the page title if not.

Even if you added five more views using the URL with the pattern, **popular/<VIEWNAME>**, only one of all of the views could be set as the default menu tab.

Look for the arrow in the following screenshot. It points to the **Parent Menu Item Title**. If **Parent Menu Item Title** is empty in the view settings, this title defaults to match the **Menu Title**. Since that was blank as well, it defaults to the **Page Title**. You could set three different titles, one each for the **Page**, the **Parent Menu Item**, and the **Menu Tab Title** (although that might lead to confusion).

Typically, both tab **Menu Tab Titles** and **Page Titles** should be the same. I do recommend filling in a new **Parent Menu Item** though. It's a bit disconcerting when **All-Time Popular content** has a parent name of **Recent popular content**. A good title might be **Popular content**.

Recent popular content					
Recent popular content All-time popular content Recent popular content					
Type	Title	Author	Created	Recent hits▼	
Audio	La Bicicletta by A Bike Commuting Trio	admin	08/04/2008 - 15:26	24	
Page	Directions	admin	05/17/2008 - 23:32	18	

12. Observe the items in the rest of the view. Consider using the Worksheet from *Recipe 10* to identify which features are used.
13. **Save** the View. If you modified the **Parent Menu Item Title**, note the change at <http://YOURSITE.com/popular> (you will see if a difference only if your theme is set to display the parent menu).

Recipe notes

- This recipe makes it look like there are many steps to enable the two statistics views. In fact, most of the steps in this recipe are simply designed to help you to understand the settings. To implement the default settings, simply follow four steps – Steps 1, 2, 5, and 6.

 Steps 3 and 4 were designed to highlight the value of **Overriding** these views, which is performed in Step 5.

- Consider doing a web search for "recent popular content" (note how many Drupal sites neglect to enable the content view count – they have empty **Recent Hits** and **Reads** fields). On the positive front, this can be a fun, quick search term if you want to find fellow Drupal developers who share your interests or live in your area. For instance:

"recent popular content" rivers

- The **Statistics** module is useful, of course, for highlighting popular site content. Also, consider using a full website statistics tool such as **Google Analytics** for much more advanced analysis of your site's usage. There is no problem with using the statistics module together with the Drupal Google Analytics module on the same site.

- For more information on the **Statistics** module, see:
<http://drupal.org/node/297>
- Sign up for Google Analytics at: <http://www.google.com/analytics>
- The Drupal **Google Analytics** module is at:
http://www.drupal.org/project/google_analytics
- If you use Google Analytics, consider becoming familiar with Google's privacy policies at: <http://www.google.com/intl/en/privacy.html>
- For technical information on Drupal's menu system, see:
<http://api.drupal.org/api/group/menu/5>

Recipe 15: All-time Popular, and Recent popular Blocks—removing the hit count

[Ingredients]
Statistics module, and Count content views enabled, as in Recipe 14.

In this recipe, we place the statistics blocks on the page. We will explore a theming modification, and explain why the theme function is necessary. We also note an alternative to using these views-created blocks to view popular content.

Adding the blocks to a region

1. Override the **popular_recent** and **popular_alltime** default views. Overriding a block is covered in *Recipe 11* (to review: simply add the default view, and save). This has the effect of also enabling the view, so you do not need to explicitly click on **Enable**.
2. Go to the **Block** configuration page. Add **popular_recent** and **popular_alltime** to the regions of your choice, and click on **Save blocks** (it's becoming increasingly fashionable to use footers for blocks such as these, though sidebars are still common). Each block include the node titles, followed by the hit or read count in parentheses.



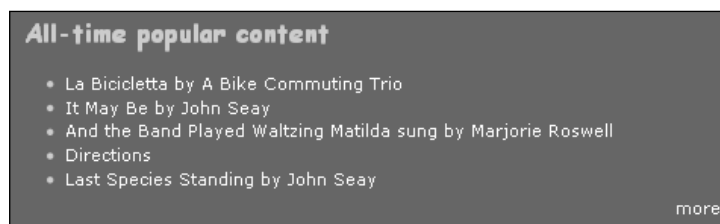
Did you notice that the two views blocks do not actually display all the fields that were so carefully designated in the Views **Fields** fieldset? (selected **Fields** included **Type**, **Title**, **Author**, **Creation Date**, and **Hits** or **Reads**). The list of fields included in the block is determined by code located in `views_statistics.inc`.

Removing the hits from the popular_alltime block

3. Logon to your host directory, and locate the `views_statistics.inc` file in the `<DRUPALROOT>/sites/all/modules/views/modules` directory.
4. Locate the `theme_views_view_list_popular_alltime` function (there is a similar function for `theme_views_view_list_popular_recent`).
5. Copy the entire function to the clipboard.
6. Paste the entire function into your `template.php` file. Change the function name, replacing the `theme` prefix with the actual name of your theme (if you are not using a version control system, make a backup of `template.php` file first!).
7. Remove the following lines:

```
if ($field['fullname'] == 'node_counter.totalcount') {  
    $item .= ' <span class="popular-content-count">  
( ' . views_theme_field('views_handle_field', $field['queryname'],  
$fields, $field,  
    $node, $view) . ')</span>';  
}
```
8. Save your `template.php` file and refresh your page.
9. Find the equivalent lines in `theme_views_view_list_popular_recent`, and repeat steps 3 through.

Your blocks will now display the node titles without displaying the hits.



Recipe notes

Astute readers may wonder: "Why did we need to invoke a special theming function here to get a list of node titles whereas no such function was needed for the **swim_group** block?" After all, in *Recipe 6* we were able to create both a full node page and a titles-only block in the same view – and we did this without any theming "magic". The difference lies in the **View Type** and whether they actually implement the field selections from the **Fields** fieldset.

View Type	Implements selected Fields?
Full Nodes	No
Teaser List	No
Table View	Yes
List View	Yes

In the statistics default views (**popular_recent** and **popular_alltime**) both the page and the block use view types that implement selected fields (the page uses a **Table View**, while the block uses a **List View**). So without a special theme function to intercept the output, both the page and the block would display all the same fields. The `views_statistics.inc` file's, `theme_views_view_list_popular_alltime` function, in essence says: "for each popular node, if the view is output as a **Block**, list just the **Title**, and – if we leave it be – the hit count" (rather than deleting hits, we could have also modified the function to add **Reads** or **Views**, helping visitors to know what the numbers mean).

The following table details which View Types are used by the blocks and pages, across three views. View Types that use field selections are noted and highlighted in grey.

	Page "View Type"	Block "View Type"
popular_recent	Table View (Uses Fields)	List View (Uses fields)
popular_alltime	Table View (Uses fields)	List View (Uses fields)
swim_groups	Full Nodes (Does not use fields)	List View (Uses fields)

Since **swim_groups** only implemented the field selection for the block, we did not need a function to distinguish the page output from the block output. An alternative to manipulating the **Fields** selection via the theme layer, is to simply create one view for the block, and another for the page. The default views didn't take that approach, hence our coverage of these special theme functions (see Chapter 7 for more on theming views).

Another alternative

In the **Blocks** administration page, that there is actually a third related block, created by the **Statistics** module, called **Popular content**. The configuration link offers **Block specific settings**.

Home » Administer » Site building » Blocks

'Popular content' block

▼ Block specific settings

Block title:

Override the default title for the block. Use <none> to display no title, or leave blank to use the default block title.

Number of day's top views to display:

How many content items to display in "day" list.

Number of all time views to display:

How many content items to display in "all time" list.

Number of most recent views to display:

How many content items to display in "recently viewed" list.

This block—along with its special configuration settings—was created by the **Statistics** module (remember that our other two **Popular** blocks were created by the `views_statistics.inc` file in the **Views** module).

Experiment with the three blocks to make a determination on whether you prefer the core **Popular content** block, or the two statistics views blocks for your site.

User-submitted snippets

- Most Popular Nodes:
<http://drupal.org/node/47570>
- Display the most popular content in a way like c|Net's what's hot:
<http://drupal.org/node/74160>

Recipe 16: Taxonomy View

Ingredients

Taxonomy module, enabled
Vocabulary with terms

The **Taxonomy** view is typically used to display a teaser list of all posts tagged with a specific term.

1. Enable the taxonomy module if it is not yet enabled.
2. If you have not yet created a vocabulary with terms on your site, go ahead and do so. Typically, vocabularies are created at `admin/content/taxonomy`, and then individual terms are added to each vocabulary. Alternatively, the **Taxonomy Manager** module, located at `http://drupal.org/project/taxonomy_manager`, is a wonderful improvement over Drupal core's Category administration. It enables administrators to search, merge, delete, move, and easily edit terms. The module was developed as a Google Summer of Code project.

The screenshot shows the 'Taxonomy Manager - Region' interface. On the left, a 'Region' list includes Southwest, Northwest, Midwest, Southeast, Northeast, and Gulf Coast (selected). The right panel displays the details for the 'Gulf Coast (51)' term. The 'Name' field is 'Gulf Coast'. The 'Description' field contains 'A new special focus on outreach in Mississippi and Louisiana'. Below are sections for 'Synonyms', 'Relations', and 'Parents', each with a text input and a '+' button. The 'Weight' is set to 5. A 'Go to the term page site' link is at the bottom.

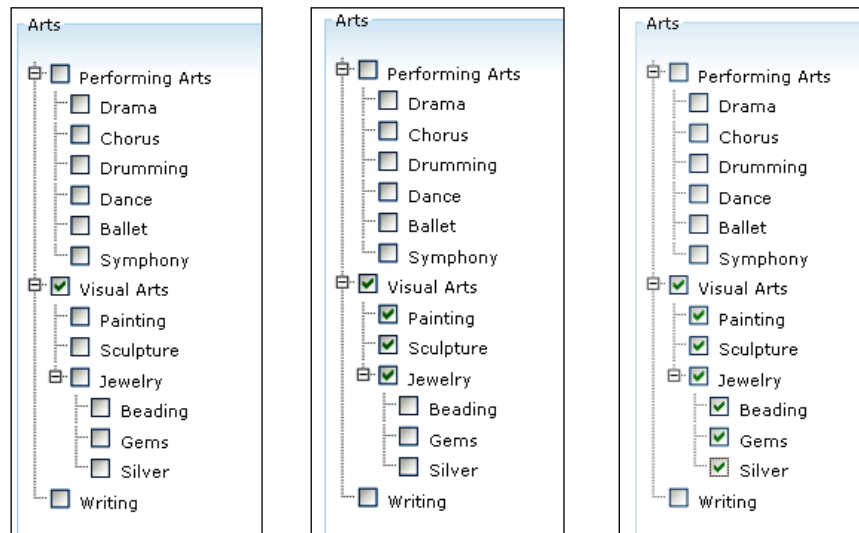
- Go to `admin/build/views`. Scroll down to the **Default Views**, and find the Taxonomy view. Note that the URL is `taxonomy/term`.

taxonomy_term	Taxonomy	The taxonomy view with a depth of 0.	Page	taxonomy/term	Disabled	Add Enable
---------------	----------	--------------------------------------	------	-------------------------------	----------	---

The URL itself seems to bring up the question: *Which term?* The answer will appear in the form of a **term ID** argument, appended to the URL. Each taxonomy term in Drupal has an entry in the `term_data` table, including a unique term ID, or `tid`. A sample page with the URL of `http://YOURSITE.com/taxonomy/term/15`, for instance, could print all nodes assigned the tag **Visual Arts**.

The view description mentions a depth property. Depth is applicable to taxonomies that have a hierarchy of terms. For the default view — which sets a depth of 0 — only items tagged **Visual Arts** will appear if the URL is `taxonomy/term/15`. A depth of 1 will include nodes tagged with **Painting**, even if those nodes are not also explicitly tagged with the **Visual Arts** term. The following table is a representation of a vocabulary. In human terms, we may think of **Beading**, **Gems**, and **Silver** as third-level terms in the **Arts** vocabulary, but with Drupal they will be accessed in a view with a depth set to 2.

Terms included when depth is set to 0	Terms included when depth is set to 1	Terms included when depth is set to 2
--	--	--



4. **Override the Taxonomy** view. Explore the options.

Description: Be sure to change the depth in this description if you modify the depth option located in the argument fieldset, below.

Nodes Per Page: A common annoyance with taxonomy views is the display of too few items per page. Don't be afraid to double or triple the default number of 10 nodes per page (or more, depending on your content).

Argument Title: One of the more interesting aspects of this view is the use of percent substitution in the title field for arguments. View titles will typically use the **Page Title** from the **Page** fieldset. However, with a taxonomy argument, you will likely want the page title to vary according to the term selected.

Arguments

Argument Type	Default	Title	Option	Wildcard	Wildcard Sub	Ops
Taxonomy: Term ID	Return Page Not Found	%1	0			Ops
The argument will filter by a taxonomy term ID. For this argument, set the option to the depth to search. See taxonomy for more information.						
Node: Feed Selector	Display All Values					Ops
This argument allows pluggable "feed" selectors. If using views_rss module, "feed" will turn the view into an RSS feed. Other modules may provide their own feeds. You may enter the title the feed will advertise in the title field here. If the feed type offers a description, enter the description of the feed in the option field here.						

For the taxonomy view, **%1** is shorthand for "substitute the appropriate term name for the first argument's term ID and make that term the title of the page". Note that the number in **%1** simply refers to the first argument (depth, by contrast, was zero-based).

How does the Views Module determine the Title of the Page or Block?

The coder-curious may wish to review `views_get_title()` in the `views.module` file, and the 'title' ops in the `views_taxonomy_inc` handler functions. These functions determine which property to use for the title of the view.

Essentially, if there is a page title, it is used — unless the view is a block, in which case the block title is used. If no block title is defined, the title will default back to the page title.



Then again, if there is an argument title — that will override either the block or the page title. Furthermore if the argument uses percent substitution, Views will actually look up the *name* of the argument (not just its ID number).

(Just to add to the titling fun, there is an error in the Views UI which notes in one place that the block title may not be blank, but several lines below, it states that the title may be blank. The latter is accurate.)

Even though the logic is sophisticated, the views titles typically behave as you would expect. If you are using arguments, as in the taxonomy view, you will get a different title, depending on the term.

Filter and Sort Criteria: The **Filter** and **Sort** fieldsets in this view are typical ones. We're listing only published nodes, and then sorting in reverse chronological order, with sticky items at the top.

Michael Kilkenny
Midwest Sculpture Visual Arts

Michael is a dynamic young graduate of the Visual Arts Outreach program and a native of Chicago. After a year studying Renaissance sculpture in Florence and Tuscany Italy, Michael has returned to share his craft with passion.

Marsha Sammise
Northeast Visual Arts

Marsha Sammise is a mother, educator, visual artist and creative free spirit. Marsha is in her 5th year with the Visual Arts Outreach program. She brings more than 20 years of experience to the program.

Recipe notes

- Consider creating URL aliases for each of your taxonomy views. **URL aliases** are managed at `/admin/build/path` (enable the core Path module to access this link). URLs with a named alias for each argument are friendlier, both to site visitors and to search engines.

Original URL <http://YOURSITE.com/taxonomy/term/1>

Aliased URL <http://YOURSITE.com/northwest>

URL aliases

Drupal provides users complete control over URLs through aliasing. This feature is typically used to make URLs human-readable or easy to remember. For example, one could map the relative URL 'node/1' onto 'about'. Each system path can have multiple aliases.

[\[more help...\]](#)

Alias▲	System	Operations
midwest	taxonomy/term/1	edit delete
southeast	taxonomy/term/2	edit delete
northeast	taxonomy/term/3	edit delete
southwest	taxonomy/term/4	edit delete
northwest	taxonomy/term/5	edit delete
gulfcoast	taxonomy/term/6	edit delete

- When you install the **Taxonomy Manager** module, be sure to find the related menus under both **Content Management** and **Site Configuration**. (When you click on a term setting such as `admin/content/taxonomy_manager/2`, there should be two columns. Depending on your administrative theme, though, you may need to scroll down to see the second column.)
- A module particularly relevant to the taxonomy view is **Views Taxargs**. This module adds new arguments to the drop-down list. The new arguments are similar to those already available in filters, but as arguments they are a bit more flexible. It is available for download at: http://drupal.org/project/views_taxargs.
- The **Taxonomy Redirect** module may also prove helpful in directing taxonomy term links to custom views. It can be downloaded from: http://drupal.org/project/taxonomy_redirect.
- **Taxonomy Hide** will hide selected vocabularies in a view. Additionally, it can also list taxonomy terms ordered by vocabulary, rather than just the default alphabetical listing of terms. This module can be downloaded from: http://drupal.org/project/taxonomy_hide.
- Given that the default taxonomy view is a teaser view, the *Read More link*, *Recipe 13* is applicable.

Recipe 17: Fixing the case of the missing Taxonomy term

There is a long-time odd bug in Views 5.x-1.6 taxonomy filtering, with a long-time patch available. In this recipe we will demonstrate the bug, and fix it.

Ingredients

A taxonomy vocabulary populated with terms

Go to `views/admin/build`, and override the **taxonomy_term** default view. Don't worry that we overrode it once already. We just need to give it a new name and new URL, along with a new Title, for good measure.

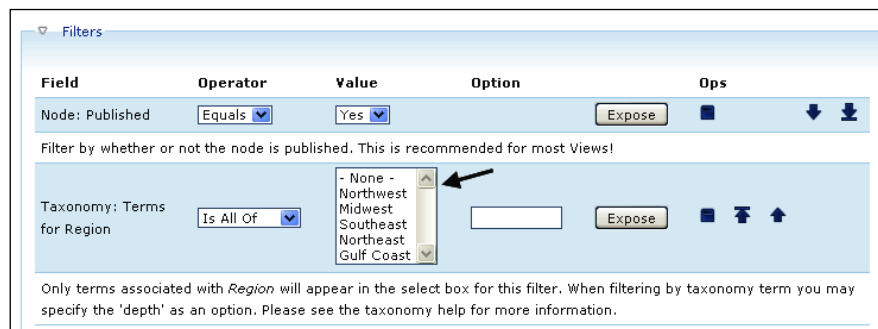
Name: `region_term`

Title: Region Taxonomy

URL: `region/term`

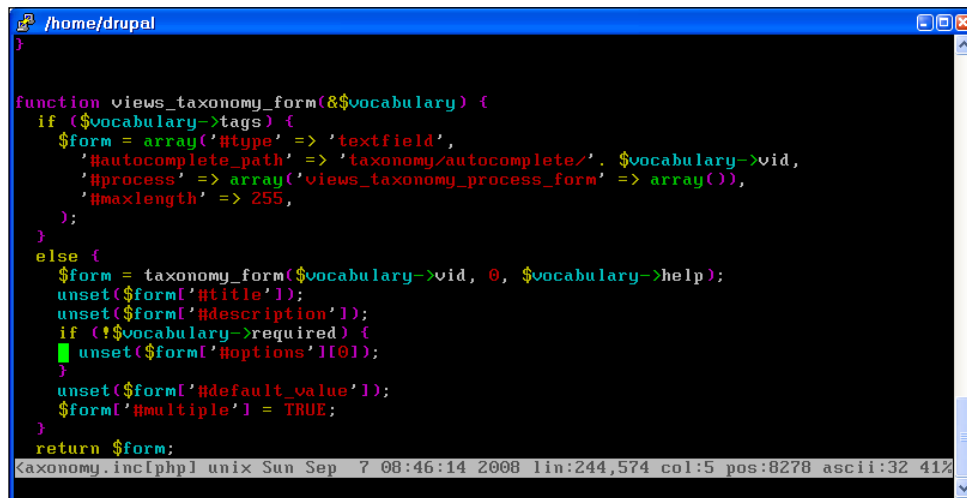
1. Add a filter such as: **Taxonomy: Terms for Region** (substitute your vocabulary name for "Region").

The first item in the list is missing. This is not always an immediately-obvious bug. You will need to know the contents of your taxonomy terms to notice this.



Many people resolve this Views 5.x-1.6 bug by adding a *dummy* term to the vocabulary, and then give it a low weight—ensuring that it's the one to be hidden. This may yield other issues. For instance, when you want to print all of your vocabulary terms, you probably don't want the dummy term in there.

- Log on to your host (if you are not using a version control system, make a backup copy of the `views_taxonomy.inc` file).
`<DRUPALROOT>/sites/all/modules/views/modules/views_taxonomy.inc`
 Open the `views_taxonomy.inc` file, and locate the `views_taxonomy_form` function.

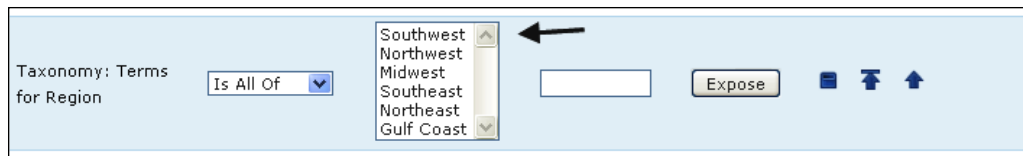


```

function views_taxonomy_form(&$vocabulary) {
  if ($vocabulary->tags) {
    $form = array('#type' => 'textfield',
      '#autocomplete_path' => 'taxonomy/autocomplete/'. $vocabulary->vid,
      '#process' => array('views_taxonomy_process_form' => array()),
      '#maxlength' => 255,
    );
  }
  else {
    $form = taxonomy_form($vocabulary->vid, 0, $vocabulary->help);
    unset($form['#title']);
    unset($form['#description']);
    if (!$vocabulary->required) {
      unset($form['#options'][0]);
    }
    unset($form['#default_value']);
    $form['#multiple'] = TRUE;
  }
  return $form;
}

```

- Change the line that reads:
`unset($form['#options'][0]);`
 to
`unset($form['#options']['']);`
- Go back to `admin/build/views/tools`, and click on the **Clear Views cache** button (the change will likely not appear to work, at first, without this step).
- Edit your view. Our edit URL is: `admin/build/views/region_term/edit`.
 Scroll down to your filters. All the terms will now appear.



Taxonomy: Terms for Region

Is All Of

Southwest
Northwest
Midwest
Southeast
Northeast
Gulf Coast

Expose

Recipe notes

This bug has served as an annoyance to many people, many of whom have reported it in various places. The appropriate place to report a Views bug is the Drupal issue queue (<http://drupal.org/project/issues/views>) — unless the bug has already been reported. If you think you have found a bug, do a concerted search to discover if the issue has already been reported. For instance, a Google search for:

site:drupal.org missing taxonomy term

yields many results. If you do a bit of investigation, you will find the authoritative issue on this subject, including a patch at this location:

Missing taxonomy terms in filters

<http://drupal.org/node/199675>

Meanwhile, if you find another issue on the same subject, it is good issue queue etiquette to mark it as a duplicate, and link to the correct previous report. Usually, when issues are reported, the patch gets rolled into a new release. In this case, the maintainers decided to wait until an underlying issue was fixed, which has not happened yet. Open source, for all its awesome strengths, does have some quirks. We will cover more topics on the Drupal issue queue in Chapter 8, to give you the tools you need for resolving such quirks.

Recipe 18: Empty Text, with PHP

[**Ingredients**]
Completed region_term view (Recipe 17)

What shall we do when the website includes generated links to taxonomy terms, but some of the terms are not yet applied? Rather than showing visitors a blank view if they click on an unused term, we will provide a more informative message. Views UI offers an **Empty Text** option into which you can place a text-only message, HTML, or PHP. In this recipe, we'll use a snippet of PHP with HTML. We will then create a taxonomy list to demonstrate the variable text.

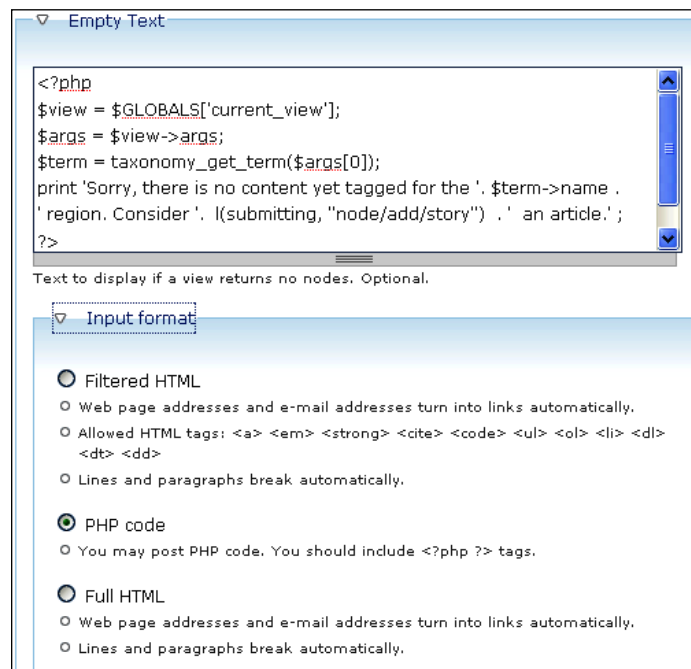
1. Open the view you created in *Recipe 17* (if you have not completed that recipe, you can still follow this recipe with any taxonomy view).
2. Open the **Empty Text** fieldset found inside the **Page** fieldset. PHP is certainly not required in the **Empty Text**. You could fill it with simple text, such as:

Sorry, there is no content tagged yet for this region

For our purposes, we will add the PHP snippet as follows:

```
<?php
$view = $GLOBALS['current_view'];
$args = $view->args;
$term = taxonomy_get_term($args[0]);
print 'Sorry, there is no content yet tagged for the '. $term->name .
' region. Consider '. l(submitting, "node/add/story") .' an
article.' ;
?>
```

Make sure to include the beginning and end PHP tag, and be sure to change the input format to **PHP code**.



Empty Text

```
<?php
$view = $GLOBALS['current_view'];
$args = $view->args;
$term = taxonomy_get_term($args[0]);
print 'Sorry, there is no content yet tagged for the '. $term->name .
' region. Consider '. l(submitting, "node/add/story") .' an
article.' ;
?>
```

Text to display if a view returns no nodes. Optional.

Input format

- ☐ Filtered HTML
 - ☐ Web page addresses and e-mail addresses turn into links automatically.
 - ☐ Allowed HTML tags: <a> <cite> <code> <dl> <dt> <dd>
 - ☐ Lines and paragraphs break automatically.
- ☒ PHP code
 - ☐ You may post PHP code. You should include <?php ?> tags.
- ☐ Full HTML
 - ☐ Web page addresses and e-mail addresses turn into links automatically.
 - ☐ Lines and paragraphs break automatically.

This code snippet gets the name of the current view and creates an array variable for the view arguments, and places the term name onto the page.


 This view has two arguments, **Taxonomy: Term ID**, and **Node: Feed Selector**. We then use a core Drupal function to get the term object for the first argument, `$args[0]`.

Generating a list of Regional categories

We will now generate a list of our categories, in order to test the empty text.

3. Go to <http://YOURSITE.com/admin/content/taxonomy>, and mouseover the edit vocabulary link. Look in the browser status bar for the vocabulary ID.

Name	Type	Operations
Region	News, Page, Story	edit vocabulary list terms add terms

4. Go to <http://YOURSITE.com/admin/build/block>, and click on **Add Block**.

Add a description of **Regions** and place the following PHP in the **Block body**. Be sure to assign the correct ID to the \$vid variable.

```
<?php
$vid = 2;
$terms = taxonomy_get_tree($vid);
print "<ul>";
foreach ($terms as $term) {
    print "<li>".
    l($term->name, 'region/term/'.$term->tid, array('title' => $term-
    >name))
    . "</li>";
}
print "</ul>";
?>
```

Block description: *
Regions
A brief description of your block. Used on the block overview page.
Block body:
<pre><?php \$vid = 2; \$terms = taxonomy_get_tree(\$vid); print ""; foreach (\$terms as \$term) { print "". l(\$term->name, 'region/term/'.\$term->tid, array('title' => \$term->name)) . ""; } print ""; ?></pre>
The content of the block as shown to the user.

5. Place the block in one of the sidebars and click on the **Save blocks** button. The **Region** vocabulary appears with a link to each term.
6. Click on a link. If no content has been assigned that term, the **Empty Text** appears, complete with the appropriate term name and a link to submit a new article.



Recipe notes

- For this exercise, if you have difficulty finding an unused term for testing the **Empty Text**, you may add a term at `/admin/content/taxonomy`. (if you did not create the term list block, make sure to note the ID for use in the `region/term/tid` URL. Take care to distinguish the vocabulary ID from the term ID).
- If your **Empty Text** contains a link to create content, be sure that the views permissions match the content submission permissions. It would be disappointing for a visitor to be invited to submit content, only to get an Access Denied message. Views permissions are set by role in the **Basic Information** fieldset, at `admin/build/views/VIEWNAME/edit`. Article submission permissions are set by role and content type at `admin/user/access`.
- A note for coders: `arg()` does not mean the same thing as `view->$args`.

Aaargh: `arg()`, and `$args`

	What is it?	What does it do?	Example
<code>arg()</code>	Core Drupal function	Return a component of the current Drupal path.	<code>arg(0)</code>
<code>view->\$args</code>	An array of arguments within the view object	Contains a list of the arguments, in the order placed in the view	<code>\$args[0]</code>

In default taxonomy views, `arg(0)` actually does not match the value of `$args[0]`. The function, `arg(0)` refers to the first component in the URL after the domain name, while `$args[0]` refers to the first view argument, which is appended to `taxonomy/term` in the URL. Note the values for a sample taxonomy default view URL, `http://YOURSITE.com/taxonomy/term/35`:

<code>arg(0)</code>	<code>taxonomy</code>
<code>arg(1)</code>	<code>term</code>
<code>arg(2)</code>	<code>35</code>
<code>\$args[0]</code>	<code>35</code>

- Helpful argument debugging code, which may be placed in the view header, is available at <http://groups.drupal.org/node/9770#comment-31673>.
- If you are curious about the functions we used in this recipe, visit the following API documentation:
 - **taxonomy_get_tree()**
http://api.drupal.org/api/function/taxonomy_get_tree/5
The **5** in the URL refers to Drupal 5; The API documentation includes functions for five different versions of Drupal.
 - **l()**
<http://api.drupal.org/api/function/l/5>
The URL character before **/5** is the letter "el." We used this function to return links to the story submission form and the taxonomy terms.
 - **arg()**
<http://api.drupal.org/api/function/arg/5>
Parse component of the URL path.
- If you are the sort who feels bogged down by code, the take-away message of this recipe is simply that you can customize the message that appears if no content is available for a given argument.

Recipe 19: Getting to know the Tracker

[	Ingredients A drupal.org account Browser with two tabs open, one for your site, and one for drupal.org	]
---	---	---

The tracker default view replicates much of the functionality of the **Tracker** module: it tracks recent posts by users.

Interestingly, the **Tracker** module does not need to be enabled for the tracker view to work. In *Recipe 21*, we will implement both the tracker view and the tracker module, together. For now, we'll just become familiar with the default view.

The Drupal website serves as a good example of the tracker:

1. Log in to drupal.org (create an account now, if you don't already have one).
2. Visit <http://www.drupal.org/tracker>, and you will see a page similar to this:

Recent posts				
All recent posts My recent posts				
Type	Post	Author	Replies	Last updated
Issue	Title and User Filter new	iem2509	2 2 new	1 sec ago
Forum topic	CSS Gallery in Drupal ! new	Roar	6 4 new	26 sec ago
Issue	Use tac_lite to hide node content but not title new	henns20	4 4 new	46 sec ago
Issue	Unable to change title of page in some situations new	jspilda	4 4 new	47 sec ago
Forum topic	javascript image path new	ssm2017	1 1 new	56 sec ago
Forum topic	Default user timezone new	sunfish	0	1 min 3 sec ago

3. Open a second browser tab for your site. Go to `admin/build/views`. Override the default tracker view by clicking on **Add**. Make note of the following as you scroll through the site.
 - **Page Information:** The URL is set to **tracker**.
 - **Page Information:** The view type is a **Table View**.
 - **Menu:** The tracker provides a menu item, called **Recent Posts**. Unlike the menus used in the two Popular posts views (Recipes 14 and 15), this menu actually appears in the website navigation menu, located generally below **My Account**, and above **Administer**.

- **Fields:** The heart of this view is the selection of fields. The table below displays the field names as they appear in the Views UI, and as they appear in the database. It also shows the column headers (labels) that are used both in the **Default Views**, and on drupal.org.

View Field	Database Table.Field	Default Tracker View Label	Drupal.org Tracker Label (for comparison)
Node: Type	node.type	Type	Type
Node: Title	node.title	Title	Post
Node: Author Name	node.author	Author	Author
Comment: Count	node_comment_statistics.comment_count	Replies	Replies
Comment: Last Comment Time	node_comment_statistics.last_comment_timestamp.	Last Post	Last Updated

Of course, you can easily change labels or add a field (the great advantage of using Views).

- **Arguments:** The view applies a single argument, a filter for the user ID. When the user ID is appended to the URL, we will see all posts by that author. Let's give this a whirl:
4. Switch back to the browser tab containing `http://drupal.org/tracker`. Mouseover **My recent posts**, and you will see your drupal.org ID number in the URL. Click to see all of your posts.
 5. You are not limited to looking at your own posts. For instance, try mousing over another user name to identify the user ID in the browser status bar. Use that number in the URL being sure to include a separator slash. Currently, drupal.org yields an **Access Denied** message if you try to use `http://drupal.org/tracker/<USERID>`. So instead, try the following URL: `http://drupal.org/user/<USERID>/track` (be sure to use a real user id).
 6. As an example, try `http://drupal.org/user/26979/track` (the example is an homage to the author of Views, known online as *merlinofchaos*). Tracker really does track! Every single post is found here.



The tracker module, covered in *Recipe 21* provides an elegant way of getting to the user URL, instead of manually entering it.

- **Arguments - alternatives:** Drupal.org tracker actually uses a different argument than the default tracker view setting.

Tracker argument	
Default tracker view	drupal.org
User: User: UID is Author	UID Authored or Commented

- Consider which approach you prefer for your site. If you would rather have posts that a user either authored or commented upon (rather than just ones they authored), click on the **delete** icon on the existing argument, and add the new one. (There is also a third option, not widely used, **Username is Author**. This is similar to **UID is Author**, but shows the user name instead of the ID number.)

Filters: Sensibly, the tracker includes only published items. Some users like to add a filter by content type, creating separate tabs for each type (use the techniques described in *Recipe 14* to create the tabs).

Sorting: By default, the tracker sorts on **Comment: Last Comment Date**. Experiment with **Comment: Last changed** for a variation.

Recipe notes

There is a terrific patch available for `views_user.inc` at <http://drupal.org/node/143356> which includes a number of helpful additional fields, filters, and sort options. For instance, you could add author email, account creation date, and/or last login to the tracker view.

Recipe 20: Making the Tracker Sortable

If you try to make each tracker field sortable, you may be disappointed with the result. If, for instance, you click on the **Title** column head, you will not get a nice alphabetical listing of all of the titles (even after setting each field in Views UI to be sortable).

Recent posts				
Type	Title ▲	Author	Replies	Last Post
Blog entry	taxonomy_term	drupal tips	0	1 day 23 hours ago
Blog entry	Drupal issue bookmark	drupal tips	0	2 weeks 1 day ago
Blog entry	Function Reference for your site	drupal tips	0	2 weeks 3 days ago
Blog entry	jQuery bookmarklet to expand all fieldsets	drupal tips	0	3 weeks 3 days ago
Page	cvs download drupal -5	drupal tips	0	3 weeks 6 days ago

The URL may look right: `http://YOURSITE.com/tracker?sort=asc&order=Title`, but the sorting is definitely wrong ("c" does not come after "j").

This is because the tracker is preset to sort by date. So, only items posted at the same time would sort alphabetically. You may modify this behavior, if you prefer.

1. Go to `views/admin/build`, and override the **tracker** default view, by choosing **Add**.
2. Give the view a new name (**sortable_posts**) and new URL (**sortable_posts**).
3. Go to the **Fields** fieldset and mark all the fields as **Sortable**. If you like, you can save the view now, and see how it works (or how it doesn't work).

Sortable	Default Sort
Yes ▼	None ▼

4. Change the **Comment: Last Comment Time** default sort from **None** to **Descending**.



A screenshot of a settings box with two dropdown menus. The first dropdown menu is set to 'Yes' and the second dropdown menu is set to 'Descending'.

5. In the **Sort Criteria** delete the **Comment: Last Comment Date** sort, by clicking on the Delete icon.
6. Save the view and open **sortable_posts**.
7. Click on the **Title** column header. The sort now works as expected.

Type	Title ▲	Author	Replies	Last Post
Blog entry	Admin Links	drupaltips	0	8 weeks 2 hours ago
Page	cvs	drupaltips	0	4 weeks 2 days ago
Page	cvs download drupal -5	drupaltips	0	3 weeks 6 days ago
Blog entry	Drupal issue bookmark	drupaltips	0	2 weeks 1 day ago
Blog entry	Function Reference for your site	drupaltips	0	2 weeks 3 days ago
Page	Handbook Hierarchy	drupaltips	0	28 weeks 3 days ago

Recipe notes

- If you have a multi-page tracker view (and you probably do), the sort will traverse pages. For instance, if you sort by clicking the **Title** column header from page 3, you will likely see the middle of the alphabet. Go back to Page 1 to see those trusty ABCs.
- Consider changing the **Menu Title** (it may no longer be appropriate to call it Recent Posts if you chose to sort alphabetically, by default, for instance).
- Tracker sorts are not case-sensitive.

Recipe 21: Integrating the Tracker View with the Tracker module

1. Enable the **tracker** module at `admin/build/modules`. It is a core Drupal module so you don't need to download anything (if the comment module is not yet enabled, go ahead and enable it since it's required for tracker).
2. Go to the **My Account** page. A link is located in your Administration menu. You may also visit **Administer | User management | Users** (`admin/user/user`), and click on a username. Note the new **Track** tab.
3. Notice the URL for this tab, `http://YOURSITE.com/user/1/track`. The 1 in the URL is a user ID. There is no connection between this URL and your tracker view. They are two different resources. You can, however, make a small modification in the URL of your tracker view enabling it to replace the module's tracker display.
4. Go to `admin/build/views` and open a **tracker** (use either the default, or one of your existing tracker views from Recipe 20 or 21).
5. In the **Page** fieldset, replace the URL with the following: `user/$arg/track`.
This matches the URL used by the tracker module. Interestingly, when you use arguments there is always an invisible `$arg` appended to the end of the view URL. For instance, we were able to use the URL `taxonomy/term/15/feed` in *Recipe 16*, even though we specified the URL as `taxonomy/term`. The URL instructions explicitly say to leave `$arg` off the end of the URL when you define the view. However, if you want an argument to appear in the middle of the URL, do add the `$arg` variable.
6. Go to the **Fields** fieldset and add **Comment: Add link**. Give it a label of **Add a Comment** (this will make it easy to distinguish the tracker view from the tracker module display).
7. **Save** the view.

8. Go to <http://YOURSITE.com/user/1/track> (you may also try a different user number). Your view now replaces the original tracker display created by the module.

recent posts for drupal tips					
View Devel load Edit Track					
View Edit Clone Export Track posts					
View successfully saved.					
Type	Title	Author	Replies	Last Post	Add a Comment
Blog entry	taxonomy_term	drupal tips	0	2 days 2 hours ago	Comment
Blog entry	Drupal issue bookmark	drupal tips	0	2 weeks 1 day ago	Comment
Blog entry	Function Reference for your site	drupal tips	0	2 weeks 4 days ago	Comment
Blog entry	jQuery bookmarklet to expand all fieldsets	drupal tips	0	3 weeks 3 days ago	Comment
Page	cvs download drupal -5	drupal tips	0	3 weeks 6 days ago	Comment

Feel free to go back and remove the **Add a comment** field, if you do not want to leave it in your tracker view.

Recipe notes

When you are logged in and viewing the tracker module pages, you will see the words **View** and **Edit** twice each. Use the links in the top row to view and edit the selected *users*. Use the links on the bottom row to view and edit the *tracker view*.

Recipe 22: Reviewing Default Views code

Ingredients

Fieldsets table from the beginning of this chapter
 Views Worksheet from Recipe 10
 Host access to the views_statistics.inc file
popular_recent view, loaded in the browser
 Statistics module, enabled
 Optional: MySQL access

In this recipe we'll pull back the curtain and examine the code behind the default views. We will not make any edits, we're just looking.

1. Open the existing **popular_recent** view at:
`http://YOURSITE.com/admin/build/views/popular_recent/edit`
Recall from Recipes 14 and 15 that the statistics module must be enabled for this view to appear.
2. Log in to your server, and open the `views_statistics.inc` file found in the `<DRUPALROOT>/sites/all/modules/views/modules` directory.
(Depending on your site configuration, it may also be found in `<DRUPALROOT>/sites/default/modules/views/modules`.
`<DRUPALROOT>` is shorthand for your Drupal install location, which may be a path such as `/home/yourname/public_html`.)
3. Find the `statistics_views_default_views()` function in `views_statistics.inc`.
The first line in the function creates a new view object.

```
$view = new stdClass();
```


The second line defines the name.

```
$view->name = 'popular_recent';
```
4. Spend some time reviewing the view object, comparing it with the views edit interface (Step 1). For this exercise it may be helpful to have your *Views Worksheet* from Recipe 10 available. Most of the connections are straightforward. For instance, the following properties correspond to the screenshot below:

```
$view->page = TRUE;  
$view->url = 'popular/latest';  
$view->page_type = 'table';  
$view->page_title = t('Recent popular content');  
$view->use_pager = TRUE;
```

☒ **Provide Page View**
 If checked this view will be provided as a page. If not checked, the fields in this group will be ignored.

URL:

 Enter the URL to use for this view in the form of 'dir/dir'. Do not begin or end the URL with a /.
 Example: 'view/tracker'. This is required if providing a page view. You can also add \$arg as a
 placeholder for arguments passed in the URL, for example 'user/\$arg/tracker' or 'node/\$arg/related'.
 Note that any arguments listed here will be required, even if they are listed as optional below. You do
 not need to list arguments at the end of the path. Do not try to use URLs such as
 taxonomy/term/\$arg.

View Type:

 How the nodes should be displayed to the user.

Title:

 The title that be shown at the top of the view. May be blank. This title ignores arguments; if you want
 your title to take arguments into account, use the "title" field in the arguments section.

☒ **Use Pager**
 If checked this query may be multiple pages. If not checked this query will be one page.

Arrays nested within the view object, correspond to fieldsets in the user interface:

```
$view->filter = array (
  array (
    'tablename' => 'node',
    'field' => 'status',
    'operator' => '=',
    'options' => '',
    'value' => '1',
  ),
);
```

Filters

Field	Operator	Value	Option	Ops
Node: Published	Equals	Yes	Expose	

Filter by whether or not the node is published. This is recommended for most Views!

There is not actually a field called "published" in Drupal, although the word appears in the user interface. When items are published, the `node.status` field is set to 1.

It may be a little disconcerting that the order of properties in the `statistics_views_default_views()` function does not match the order of items in the Views user interface. Explore to your heart's content, until you feel comfortable with the view object.

5. (Optional) Login to your MySQL database and locate the following tables installed with the Views module:

```
view_argument
view_exposed_filter
view_filter
view_sort
view_tablefield
view_view
```

The primary table is `view_view`.

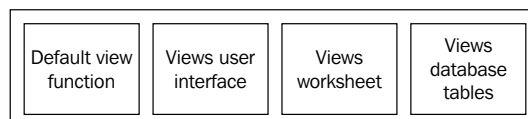
A `view_view` table excerpt

Browse Structure SQL Search Insert Export Import Operations Empty Drop									
☐	Field	Type	Collation	Attributes	Null	Default	Extra	Action	
☐	vid	int(10)		UNSIGNED	No	0			
☐	name	varchar(32)	utf8_general_ci		No				
☐	description	varchar(255)	utf8_general_ci		Yes	NULL			
☐	access	varchar(255)	utf8_general_ci		Yes	NULL			
☐	page	int(1)			Yes	NULL			
☐	page_title	varchar(255)	utf8_general_ci		Yes	NULL			
☐	page_header	longtext	utf8_general_ci		Yes	NULL			
☐	page_header_format	int(4)			No				
☐	page_empty	longtext	utf8_general_ci		Yes	NULL			
☐	page_empty_format	int(4)			No				
☐	page_footer	longtext	utf8_general_ci		Yes	NULL			
☐	page_footer_format	int(4)			No				
☐	page_type	varchar(20)	utf8_general_ci		Yes	NULL			
☐	use_pager	int(1)			Yes	NULL			
☐	nodes_per_page	int(5)			Yes	NULL			
☐	url	varchar(255)	utf8_general_ci		Yes	NULL			

Once again, the order of fields in `view_view` does not always match the default view function *or* the view's user interface. However, the naming conventions are straightforward, so it is generally not too hard to figure out the correspondences.

Recipe notes

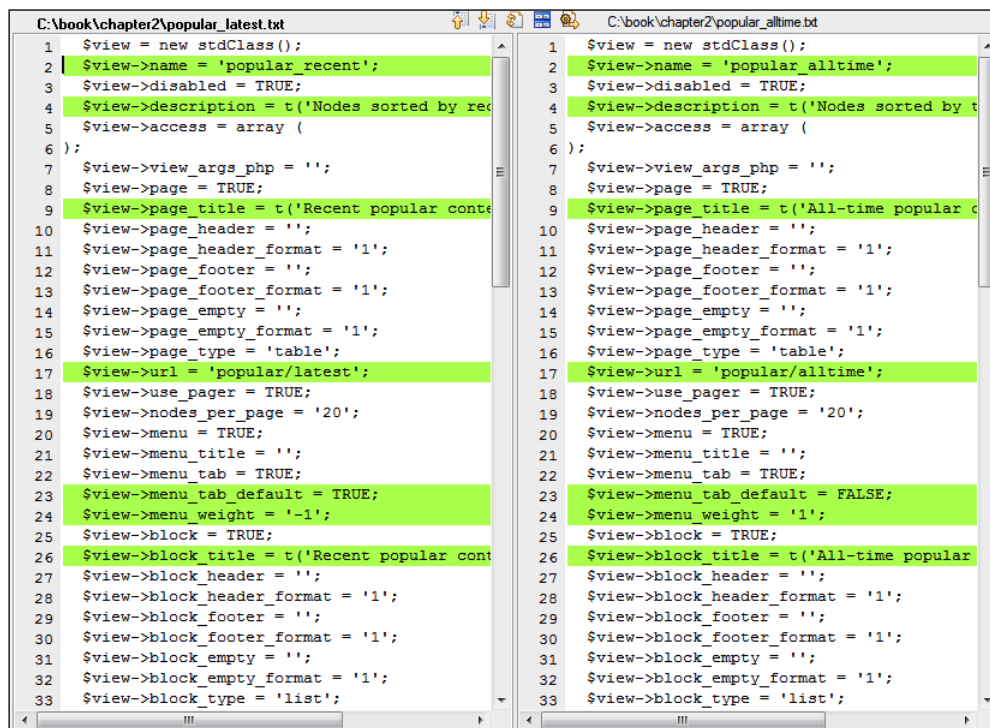
- In this recipe we explored up to four items, in order to get a better understanding of the view object:



- Kent Bye from the Lullabot company produced a beautiful graphic linking the Views user interface to the object properties and the database fields for a particular view:

<http://www.flickr.com/photos/kentbye/511840447/sizes/o/>

- When comparing two different default views, it may be helpful to use your editor's "diff" function. For the comparison that follows, I copied the two view objects from `views_statistics` into two separate files (selecting a text range from `$view = new stdClass();` down to `$views[$view->name] = $view;`. I named one file **popular_latest.txt**, and the other **popular_alltime.txt**.



Now it is easy to see what is different, and what is the same, between the two views. This screenshot features **PSPad**, a little-known, free Windows editor with a colorful file comparison. Many other applications have similar functionality, including **Eclipse** and **diffMerge**.

Recipe 23: Creating a Default View

[Ingredients]
Completed Recipes 4 and 6

In this recipe we will create a module to implement a default view. At the core of this recipe is a Views hook called `hook_views_default_views()`. This should be a fairly accessible recipe, even if you don't consider yourself to be an accomplished coder. The module will have three files.

- `swim_groups.info`
 - `swim_group.views_default.inc`
 - `swim_groups.module`
1. If it isn't there already, create a new directory under `<DRUPALROOT>/sites/default/modules/` called **custom** (of course, if the `modules` directory doesn't yet exist, make sure to create that first).
 2. Create a directory under `<DRUPALROOT>/sites/default/modules/custom` called `swim_groups`.

The ".info" file

3. Create a file in `<DRUPALROOT>/sites/default/modules/custom/swim_groups`, named `swim_groups.info`, containing the following content:

```
; $Id$
name = Swim Groups
description = Module demonstrating a default view
dependencies = views
package = Other
```

Default Views function in a ".inc" file

4. Create a file named `swim_groups.views_default.inc`. Include the following code:

```
<?php

/**
 * Implementation of hook_views_default_views()
 */

function swim_groups_views_default_views() {
}
```

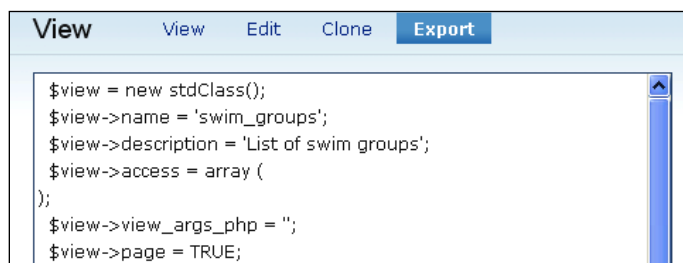
In naming our function, we are abiding by the Drupal standard of `module_name_hookname`. The hook for default views is called `hook_views_default_views`, hence the name, `swim_groups_views_default_views`.

By creating a separate file `swim_groups.views_default.inc`, we are adopting a practice that has become standard in Drupal 6, but that is seldom used by Drupal 5 developers. Most module authors for Drupal 5 place `hook_views_default_views` either in the `.module` or an `.inc` file, together with other functions. That is certainly a valid approach. It is helpful, though, to be able to know – just by seeing the filename – that a default view is available for a given module. For a module containing more than just a few files, we would place this file in a separate `includes` subdirectory.

5. Switch to the browser, and edit the **swim_groups** view
http://YOURSITE.com/admin/build/views/swim_groups/edit.

If you do not have the **swim_groups** view available, another view will work as well for the purpose of this recipe.

6. Click the **Export** button (an **Export** button is also available at `admin/build/views`).



The **Export** button creates most of what you need for the default view function!

7. Simply select the text, and copy it into `swim_groups.views_default.inc`, just below the function name.
8. There are just a few things that you should add to the function.
 - **The Translate function:** It is good practice to run title and label strings through the translate function. Edit the description and page and block title as follows:


```
$view->description = t('List of swim groups.');
```

```
$view->page_title = t('Swim Groups');
```

```
$view->block_title = t('Swim Groups');
```
 - The view name and the URL should not be run through the `t()` function.

- **Disabled Property:** Add the following line just after the view name:

```
$view->disabled = TRUE;
```

This is a Boolean flag. TRUE indicates that the View should be disabled by default.

- **Return the View:** Just before the closing bracket, add a return statement.

```
return $views;
```

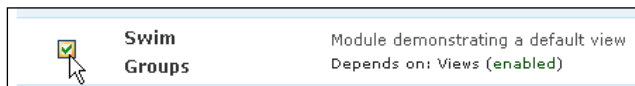
Including the Default View in a ".module" file.

9. Create a text file in the swim_groups directory called swim_groups.module. Add the following to the file:

```
<?php
// $Id$

/**
 * @file
 * Include the swim_groups default view
 */
if (module_exists('views')) {
  require_once './'. drupal_get_path('module', 'swim_groups')
    .'/swim_groups.views_default.inc';
}
```

10. Enable your module at admin/build/modules.

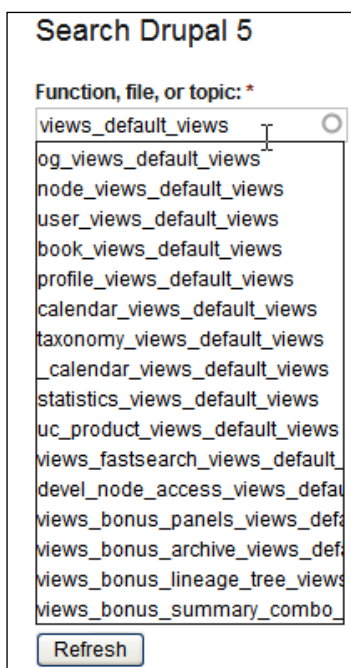


Go to admin/build/views. Your default view is successfully installed. If you wish, you may delete the **swim_groups** view that's listed under **Existing Views**. Our default view status will change from **Overridden** to **Disabled**. Of course, now you may Add the view again.

swim_groups	New Menu	List of swim groups	Page, Block, Menu	swim-groups	Disabled	Add Enable
-------------	----------	---------------------	-------------------	-------------	----------	------------

Recipe notes

- The `.info` file is not PHP so it does not use opening or closing php tags.
- The `.inc` file, and the `.module` file both begin with an opening `<?php` tag. Do not add the closing `?>` at the end of your Drupal code files.
- The Contributions API site http://api.freestylesystems.co.uk/api/function/statistics_views_default_views/5 is a terrific resource, enabling you to search and review all of the functions in selected contributed modules.



- Any changes you make to an overridden view will be stored in the database, not in code. Once you have finalized the preferred view, you may update the `.inc` file.
- Drupal developer Michelle Cox offers this helpful thread on managing views in code: <http://shellmultimedia.com/articles/managing-your-views-code>. If you choose to implement this, you will gain the ability to track your views with a versioning system such as **Concurrent Versions System (CVS)**.
- Module writers may appreciate Bevan's Config to Code module, <http://drupal.org/project/c2c>, as well as ComputerMinds' code snippet providing additional support for managing views export files. <http://www.computerminds.co.uk/organising-your-views-views-1-x>.

Summary

This chapter has covered a lot of ground. We implemented six widely-used default views that come with core Drupal, and created our own default view in code. We implemented views features including different menu types, column sorting, and arguments. We had a chance to see the source code for the default views, and make a few modifications, including a bug fix and a theme function override. Consider using your *Views Worksheets* from *Recipe 10* to identify the features of each view. Check *Appendix A* for hundreds of additional default views made available by contributed modules.

3

CCK and Views

Views and CCK go together like hand and glove. CCK is the awesome **Content Construction Kit** module, which enables the easy creation of Drupal content types. Below is a brief list of selected CCK field types that can be incorporated into whole content types. *Appendix B* includes a more extensive list of CCK field modules. The full list of CCK-related modules is found at:

<http://drupal.org/taxonomy/term/88>. or [http://drupalmodules.com/module-finder?c=Content%20Construction%20Kit%20\(CCK\)&v=5.x](http://drupalmodules.com/module-finder?c=Content%20Construction%20Kit%20(CCK)&v=5.x) (the second URL selects only those modules that apply to Drupal version 5.x).

Sample CCK field types

(*Field types that come with The CCK module are starred below)

Email	Node Reference*
Field Group	Number*
Flashcard	Text*
Imagefield	User Reference*
Link	Viewfield

Recipe 24: Addressing the Views Body Field issue

Ingredients

CCK module: <http://drupal.org/project/cck>

Number field module, included in CCK module

Text Field, included in CCK module

Field Copy module: http://drupal.org/project/field_copy

Part of CCK's magic is that created fields are automatically available in the Views fields and argument fieldsets. Additionally, developers can write code to make CCK fields sortable and filterable by Views. In essence, CCK field modules give you all the tools you need to enter content, and display it through Views.

There is, however, a commonly discovered quirk when using CCK with the body field. This recipe will demonstrate the quirk, and offer a solution for managing it.

Demonstrating the Body Field issue

1. Add two fields to the **swim_group** content type at **Administer | Content management | Content types | Swim Group | Add Field** (`admin/content/types/swim-group/add_field`). It is good practice to prepend the name of the content type (or an abbreviated name) to the fieldname. Doing so makes it easier to find the right field when creating a view.

Field name	Field type	Label
swim_group_instructor	Text: Text field	Instructor
swim_group_minimum_age	Integer: Text field	Minimum Age

2. Populate the **Instructor** and **Minimum Age** fields with content.
3. Clone the **swim_groups** view at **Administer | Site Building | Views | Clone** (`admin/build/views/swim_groups/clone`). Make the following edits.

Name	swim_groups_table				
URL	swim-groups-table				
View Type	Table View				
Fields	Add Node: Body and Text: Instructor . Provide labels for all three fields (Class , Description , and Instructor).				

Name	Label	Handler	Option	Sortable	Default Sort
Node: Title	Class	Normal	As link	No	None
Display the title of the node.					
Node: Body	Description	Full Text			
Display the Main Content.					
Text: Instructor (field_swim_group_instructor)	Instructor	Group multiple values	Default	No	None

Sort Criteria	Minimum Age, Ascending
----------------------	------------------------

4. Observe the view at <http://YOURSITE.com/swim-groups-table>.

Class	Description	Instructor
Water Wrigglers (Ages 2 to 3)	minimum_age: 2 Instructor: Katie Rossow For those new to the water. Basic skills such as floating, kicking, and blowing bubbles. The group is led by professional child educators who are also certified water instructors	Katie Rossow
Tadpoles (Ages 3 to 5)	minimum_age: 3 Instructor: Ilana Papava Basic water fundamentals, including floating, kicking, paddling, and breath control.	Ilana Papava



Unfortunately, the **Node: Body** field has a special property: it appears to contain the full node instead of just the expected body field.

In general, I recommend not using the **Body** field at all in a CCK content type (you may simply delete the **Body** field at the Add View screen, <admin/content/types/add>).

Title field label: * Title
Body field label: Body
To omit the body field for this content type, remove any text and leave this field blank.

However, what are we to do in this case, when the **Body** field already exists and is already populated? Essentially, we will create a new CCK **Description** field and copy the **Body** text into it. Then, in the View, we will replace the **Body** field with the new **Description** field.

Copying the Body Field, and modifying the View

5. Add a **Description** field (**swim_group_description**) field to the swim_group content type, at admin/content/types/swim-group/add_field.
6. Install the **Field Copy** module located at: http://drupal.org/project/field_copy. Enable the module at admin/build/modules.
7. Perform the body field copy at admin/content/field_copy.

swim_group

source: target: Nullify Source: ☐

8. It is a good idea to ensure that the copy was actually performed by either pulling up some records, or viewing them in the database. This is a view of the **content_type_swim_group** table, in phpMyAdmin.

←T→	vid	nid	field_instructor_value	field_minimum_age_value	field_swim_group_description_value
☐	2	2	Katie Rossow	2	For those new to the water. Basic skills such as f...
☐	3	3	Ilana Papava	3	Basic water fundamentals, including floating, kick...
☐	4	4	Ilana Papava	6	Front and back float and paddle stroke. Rotary bre...
☐	6	6	Sara Zlatnick	15	Learn to swim during a special hour with no childr...
☐	5	5	Michael Phelps	9	Introduction to back crawl, diving, treading water...

9. Once you are satisfied that the field has successfully copied, you may repeat the field copy again, at admin/settings/field_copy. This time with the **Nullify Source** item checked.
10. Edit the view at admin/build/views/swim-groups-table/edit, and replace the body field with the new **Swim Group Description** field. Use the arrow keys to control the order in which the fields appear.
11. Observe the view at <http://YOURSITE.com/swim-groups-table>. The body quirk is now resolved.

Class	Description	Instructor
Water Wrigglers (Ages 2 to 3)	For those new to the water. Basic skills such as floating, kicking, and blowing bubbles. The group is led by professional child educators who are also certified water instructors	Katie Rossow
Tadpoles (Ages 3 to 5)	Basic water fundamentals, including floating, kicking, paddling, and breath control.	Ilana Papava

Recipe notes

- Install the **Backup and Migrate** module available at: http://drupal.org/project/backup_migrate, to backup your Drupal database before making changes. Alternatively, you can use `mysqldump` at the command line.

```
mysqldump -u MYUSERNAME -p MYDBNAME > backup.sql
```

- **Backup and Migrate** offers the benefit of not backing up unneeded cache files. Make sure to backup the database to a directory that is not on the internet, i.e. do not leave the SQL output file in your `public_html` directory or any of its subdirectories.
- Without the **field_copy** module you may be able to perform the following SQL in your database to achieve the same result.

Copy the **Body** found in the node revisions table to the **content_type_swim_group** table:

```
UPDATE content_type_swim_group ct, node_revisions nr
SET    ct.field_swim_group_description_value = nr.body
WHERE  nr.vid = ct.vid
```

Now clear the **body** and **teaser** from the **node_revisions** table:

```
UPDATE node_revisions nr, content_type_swim_group ct
SET    nr.body = "", nr.teaser=""
WHERE  ct.vid = nr.vid
```

- Neither approach (the **field_copy** module, or the SQL above) will work if the `content_type_<CONTENTTYPE>` table is not already populated. If the above approaches fail, you can always default to manually copying the field in when you edit in Drupal.
- Consider installing the Views Sort helper module (http://drupal.org/project/view_ui_sort), written by the same author as the helpful **Field Copy** module. Views Sort provides an easier-to-browse drop-down list for fields and arguments in the Views UI. The module's modified sort order is especially appealing if you use a `contentname_fieldname` convention for naming fields.

Recipe 25: Creating Website snapshots

Ingredients

Content module: <http://drupal.org/project/cck>

Link module: <http://drupal.org/project/link>

CCK Extras module: http://drupal.org/project/cck_extras

This recipe makes use of the **Link** and **CCK Extras** modules to take mini-snapshots of websites. CCK Extras includes this enjoyable description in the `README.txt` file:

CCK offers a bunch of swanky hooks for editing and displaying the contents of assorted fields. CCK Extras is a module that adds a few convenient (but sometimes esoteric) formatters.

Essentially, a formatter function in the CCK Extras module takes the URL from the link field as input and delivers a site screenshot as the output.

Installing modules

1. Install and enable the modules listed in the ingredients section.

Creating a Content Type

2. Create a new content type for links at `admin/content/types/add`.

Name	Link
Type	link
Description	URL
Default Options	Check Published
	Uncheck Promoted to the frontpage

3. Select **Save content type**. This brings you back to `admin/content/types`.

Adding the Link Field

4. Edit the new **Link** content type.
5. Click on the **Add Field** tab. Set the **Name** and **Field type**.

Name	link
Field Type	Link: Text Fields for Title and URL

Field type: *

Link

☒ Text Fields for Title and URL

6. Click the **Create field** button. This brings you to: `admin/content/types/link/fields/field_link`.
7. Most of the fields at field administration page may remain as their defaults. Consider changing the following fields:

Label	Link (<i>same as original, but capitalized</i>)
Help Text	Full URL, including: http://
Link Title	No Title

8. Click The **Save Field** settings button.

Create a Website Snapshots View

9. Create some content items at `node/add/link`.
10. Create a new view at `admin/build/views/add`.

Name	website_snapshots
Description	Website snapshots, using the websnapr service
Provide Page View	Checked
URL	website_snapshots
View Type:	List view
Title:	Website Snapshots
Use Pager	Checked
Nodes Per Page	15

Fields**Node Title - Option: Without Link****Link - Option: URL, as link****Link - Option: Preview image (small)**

Fields

Name	Label	Handler	Option	Sortable	Default Sort
Node: Title		Normal v	Without link v	No v	None v
Display the title of the node.					
Link: Link (field_link)		Group multiple values v	URL, as link v	No v	None v
Link: Link (field_link)		Group multiple values v	Preview image (small) v	No v	None v

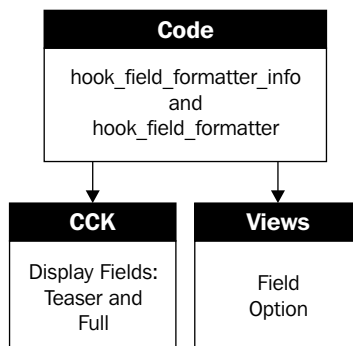
Filters**Node: Published Equals Yes****Node: Type Is One Of Link****Link: Link - Protocol Is One of http, https****Sort Criteria****Node: Title, Ascending**11. Click on **Save** and enjoy the view at:http://YOURSITE.com/website_snapshots.

While the screenshot is being prepared, it may temporarily look like this:



Recipe notes

- Find more information about **Websnappr** at <http://www.websnappr.com/>.
- A related Drupal module is also available at http://gestaltware.com/projects/websnappr_drupal.
- The **Option** dropdowns in the Views **Fields** fieldset are getting information from CCK's field formatter functions. In the recipe above, we formatted the same field twice, implementing the **URL** in both, as **link** and the **Preview image (small)**, formats.
- The same formats appear in both the CCK and Views administrative screens. The following diagram is a summary view of more detail.



- The `hook_field_formatter_info()` function provides basic information about each format including the name and the field type to which it applies. A second function, `hook_field_formatter()` goes on to provide the actual formatting. In the table below, we see that two modules combine formatters from the **Link** and **cck_extras** modules to provide a total of nine formats (additionally, the CCK module provides the "Hidden" format).

Field formatters from link module	Field formatters from cck_extras module (excerpt)
<pre>function link_field_formatter_info() { return array('default' => array('label' => t('Title, as link (default)'), 'field types' => array('link'),), 'url' => array('label' => t('URL, as link'), 'field types' => array('link'),), 'plain' => array('label' => t('URL, plain text'), 'field types' => array('link'),), 'short' => array('label' => t('Short, as link with title "Link"'), 'field types' => array('link'),), 'label' => array('label' => t('Label, as link with label as title'), 'field types' => array('link'),), 'separate' => array('label' => t('Separate title and URL'), 'field types' => array('link'),),); }</pre>	<pre>function cck_extras_field_formatter_info() { return array('redirect' => array('label' => 'Link redirection', 'field types' => array('link'), 'module' => 'cck_extras',), 'link_snappr_tiny' => array('label' => t('Preview image (tiny)'), 'field types' => array('link'), 'module' => 'cck_extras',), 'link_snappr_small' => array('label' => t('Preview image (small)'), 'field types' => array('link'), 'module' => 'cck_extras',),); }</pre>

Note how the code corresponds to the Drupal user interface.

In the CCK **Display fields** administration, these combined formats appear under **Teaser** and **Full** at `admin/content/types/link/display`. The teaser could be set to show the URL as a link, for instance, while the full node displays it as an image.

**CCK Display fields
Teaser and Full drop-down items
created from field formatters**

Field	Type	Label	Teaser	Full
Link	Link	Above	Title, as link (default)	Title, as link (default)

Submit

In the Views administration, the formatters appear in the **Fields** fieldset under **Option**.

**Views Option drop-down items
created from field formatters**

Name	Label	Handler	Option
Node:		Normal	Without link
Title			
Display the title of the node.			
Link: Link (field_link)		Group multiple values	URL, as link
Link: Link (field_link)		Group multiple values	Preview image (small)

While the word "formatter" does not actually appear in the Drupal administration screens, it is a helpful word to know for writing your own code and when looking up information on drupal.org.

- `hook_field_formatter()`, along with the associated `hook_field_formatter_info()` are deprecated in Drupal 6 in favor of standard theme functions.

Recipe 26: Creating a Blogroll

[Ingredients
Completed Recipe 25]

In this recipe we will clone the **website_snapshots** view, and make some modifications, to turn it into a blogroll, which is a block of links to great sites.

Modifying the Link Field

1. Administer the link field at `admin/content/types/link/fields/field_link`.
2. Change the help text to read:
Link Title Example: `<em>Local Food</em>`
`<br>`Link URL Example: `<em>http://en.wikipedia.org/wiki/Local_food</em>`
`<br>`Note: Use the full URL, including: `<em>http://</em>`
3. Change the **Link Title** from **No Title** to **Required Title**.

Link Title:
☐ Optional Title
☒ Required Title
☐ Static Title:
☐ No Title

4. Go to `admin/content/node` and **edit** each link item. Enter a **Link Title** for each content item.

Site Name: *

Link Title: **Link URL:**
Link Title Example: *Local Food*
Link URL Example: *http://en.wikipedia.org/wiki/Local_food*
Note: Use the full URL, including: *http://*

Creating a new view based on `website_snapshots`

5. Go to `admin/build/views/website_snapshots/edit`, and click on **Clone**. Make the following modifications:

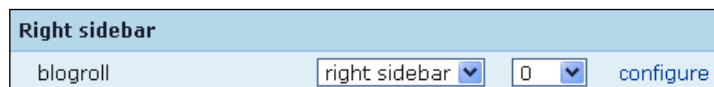
	From	To
Name	<code>website_snapshots</code>	<code>blogroll</code>
Description	Web site snapshots, using the websnapr service	Block of links to great sites
URL	<code>website_snapshots</code>	<code>blogroll</code>
Provide Block		Checked
View Type		List View

	From	To
Title		Blogroll
Nodes Per Block		30
[More] Link?		Checked
Fields	Node Title	Delete this field
	Link: Link (field_link)	Change Option to: Title, as Link (Default)
	Link: Link (field_link) that had been formatted as a Preview	Delete this field
	Image (small)	

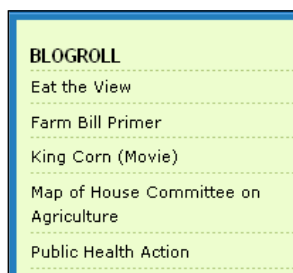
6. Save the view.

Placing the Blogroll Block

7. Place the block at `admin/build/blocks` into one of the sidebars, and save the block.



8. Enjoy the View.



Recipe notes

- It may be tempting to try Drupal's blogroll module. However, a brief look at the issue queue <http://drupal.org/project/issues/blogroll> yields active issues associated with this module that are mostly more than four years old. The blogroll module may be considered deprecated primarily because Views, together with the Link module, does such a good job of providing blogroll functionality.

- It is certainly possible to make a blogroll without starting from *Recipe 25*. This recipe offered the chance to review features such as cloning a view, and modifying the link widget and formatter options.
- A web search for "blogroll etiquette" yields some helpful discussion on how many and which links to include, whether to ask permission to link, whether to ask why your site was removed from someone else's blogroll, and whether or not to share that you are deleting a site from your blogroll.

Recipe 27: Displaying Guitar Chords

Ingredients

Content module: <http://drupal.org/project/cck>

Guitar module: <http://drupal.org/project/guitar>

Field formatters—such as the link formatters used in the two previous recipes—are available for a wide variety of modules. Musicians may be glad to know of a field formatter in Drupal that generates guitar chord diagrams.

1. Ensure that the CCK **Content** module is enabled in `admin/build/modules`.
2. Enable the **Guitar Chord Diagram Generator** and **Guitar CCK Field** modules that come with Guitar. You may also choose to enable the Guitar input filter.
3. Create a new Chord content type at `admin/content/types/add`, and add a field called **chord** at `admin/content/types/chord/add_field`.
4. Populate the content type at `node/add/chord`. The Chord Name may match the **Node Title**.

Chord Name:

Examples: C major, Bb minor 7th, F# augmented.

Notes:

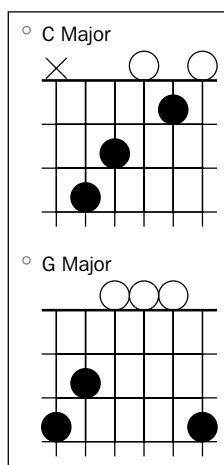
A comma-separated list of numbers, indicating the fret to be played on the instrument. Use 'x' for strings that are not played.

Notice that the **Notes** field calls for a comma-separated list of numbers indicating the pressed frets on the guitar.

5. Create a View at `admin/build/views/add`.

Name	chords
Description	Guitar chords chart
Provide Page View	Checked
URL	chords
View Type:	List view
Title:	Chords
Fields	Node: Title Guitar Chord: chord (field_chord) <i>Note that there is only one formatter in the Option selection list: Default</i>
Filters	Node: Published Equals Yes Node: Type Is One Of Chord
Sort Criteria	Node: Title Ascending

6. Enjoy the view.



Recipe notes

- *Appendix II* contains an extensive list of additional field formatters, including Amazon product fields, flashcards, Flickr sets, Google maps, podcasts, star rating systems, international zip codes, and more.
- The **Guitar Chord Finder** will aid in determining guitar chord notation (<http://www.cyberfret.com/chords/chord-finder>). Scroll about halfway down the page. Click on a chord name to display the notation. Be sure to add the commas when including these notations with the Drupal Guitar module.

Recipe 28: Using the Viewfield module

Ingredients

Two CCK content types

An existing view based on one of the content types

Viewfield module: <http://drupal.org/project/viewfield>

Token module: <http://drupal.org/project/token>

Taxonomy module: Comes with core Drupal

Ubercart module and dependencies (optional): <http://drupal.org/project/ubercart> or <http://install.ubercart.org/>

Viewfield is a CCK field type. It enables the contents of a view to be displayed in a node. In this recipe, we will display a view containing a list of products in a swim class node.

Installing modules

1. Install and enable the modules listed in the ingredients (**Ubercart** installation is optional. If you do install it, note the many module dependencies listed on the project page. You may also choose the Uberinstaller package for a full Drupal installation, plus Ubercart and all its dependent modules at once. Setup of Ubercart, a shopping cart program used to create the products, is beyond the scope of this book.)

Views, Taxonomy, and Content setup

2. Identify the taxonomy view you wish to use for this exercise. In the example we will use a page view called **products**, which displays a table with three fields **Node: Title**, **Product: Sell Price**, and **Product: Buy it now button**. We'll filter for the **Product** content type, and implement a **Taxonomy: Term Name** argument. (If you do not wish to install Ubercart, simply create a content type called Product, apply taxonomy terms to it, and create a taxonomy view. A taxonomy view uses a term or term ID as an argument. See Recipes 16-18 for more detail on taxonomy views.)
3. By sharing vocabulary terms across multiple content types, we can easily link nodes to related views. Ubercart comes with a vocabulary named Catalog. By default, it is applied only to the Product content type. Be sure to enable the assignment of Catalog terms to the Swim Group content type by clicking on the **edit vocabulary** button at `admin/content/taxonomy` (if you are using your own content type rather than Ubercart's Product content type, make sure that it has an associated vocabulary as well).



Types: *

☐ Page

☒ Product

☐ Story

☒ Swim Group

- Click on **Add Terms** for the Catalog vocabulary at `admin/content/taxonomy`. Add the term name: **swimming**. Since the Catalog vocabulary has a hierarchy of terms enabled, you will also need to select the `<root>` as the parent term.



Catalog

Parents:

`<root>`

crafts

photography

Parent terms.

Term name: *

swimming

The name of this term.

- Create several products (with or without Ubercart installed), and assign them the term **swimming**. Likewise, edit some of the swim groups and assign them the term **swimming** as well.
- Now we will add the Viewfield to `swim_group` at `admin/content/types/swim_group/add_field`. Name the new field **View**, and choose the Viewfield **Select List** widget.



View field

☒ Select List

Create field

- Click on the **Create Field** button. This brings up more Viewfield configuration options. Here we specify the associated view along with its arguments.

Label	Product links, for your convenience
Allowed views	Check Products
Use a common default value for all nodes if the user does not override it on the node form.	Checked
View dropdown	Products
Arguments	swimming (or [term])



If both content types (in this case **swim_group** and product) share the same taxonomy, it is appropriate to replace the hard-coded argument (such as **swimming**) with a token such as **[term]**. In this way, the product view's details can vary depending on the term.

Associating the View with a Content item

8. Edit an existing **swim_group** content item, for instance, `node/4/edit`. You will now have the option to select a view along with associated arguments. If you want the entire view to display, you may leave off the arguments option. In this case, we're using the **[term]** token so the products sharing the same taxonomy term as the content will appear.

Product links, for your convenience

☒ Override default

Product links, for your convenience:

products

Arguments:

[term]

Provide a comma separated list of arguments to pass to the view. These arguments will be passed to EACH selected view. If an argument contains commas or double quotes, enclose it in double quotes. Replace double quotes that are part of the argument with pairs of double quotes. Use the syntax [token] if you want to insert a replacement pattern.

Replacement patterns

Token	Replacement value
Node tokens	
[nid]	Node ID
[type]	Node type
[type-name]	Node type (user-friendly version)
[title]	Node title
[title-raw]	Unfiltered node title. WARNING - raw user input.
[author-uid]	Node author's user id
[author-name]	Node author's user name
[author-name-raw]	Node author's user name. WARNING - raw user input.
[term]	Name of top taxonomy term

- Click on the **Save** button. Enjoy the view that's now linked to your content item.

Flying Fish (Ages 9 - 14)

Introduction to back crawl, diving, treading water and improving front crawl and rotary breathing skills.

Product links, for your convenience:

Floral Embossed Bathing Cap	\$10.00	Add to cart
Nose Clips and Ear Plugs Set	\$3.00	Add to cart
Swim Goggles	\$13.25	Add to cart

Recipe notes

- Some additional use cases for Viewfield include: albums with associated tracks, classes with required reading assignments, committee descriptions with associated meeting minutes, or events with associated photographs.
- If you wish to continue to use the original **swim_groups** view, it may be preferable to change the type from a **Full node** to a **Teaser view** so that the same products don't display multiple times. Turn the display of the products off for teasers at `admin/content/types/swim-group/display`.

Teaser

<Hidden>

- By default, Viewfield implements the Page settings. You may choose the Block settings instead, at the **Display fields** tab, `admin/content/types/<CONTENTTYPE>/display`.

Full

Table View

Use view "Page" settings

Use view "Block" settings

Count of items in view

Full Nodes

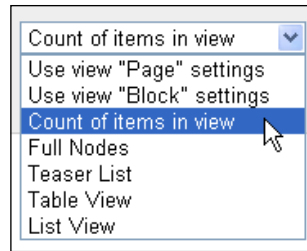
Teaser List

Table View

List View

<Hidden>

- A content type may include more than one Viewfield. For instance, one Viewfield could include a product listing, while another displays a view of instructors. In both cases, the views are associated with the content item's taxonomy.
- When setting up a Viewfield, for instance, at `admin/content/types/swim-group/fields/field_view`, consider if you want to check **Force Default**. If this is checked, the user will not be shown the Viewfield options on the edit node page. One of the authors of this functionality dubbed this approach "stealth views".
- A powerful Viewfield feature worth exploring is the option to nest a view inside another view. For instance, edit the **swim_group** view. In the **Fields** fieldset, add the **Viewfield**. Dropdown **options** for a Viewfield appear in the figure below. These are naturally the same formats found in the Display Fields options. The **Count of items in view** is a popular choice.



- These are the same options as in the CCK Display fields, but without the **<Hidden>** option. To hide the display, just delete the field from the view.
- It's a bit amusing to see that this module was launched with discussion on the tongue-in-cheek "Drupal for Evil" group: <http://groups.drupal.org/node/1826>. The creation of this group was, no doubt, a diabolical foil to the previously created "Drupal for Good" group <http://groups.drupal.org/drupal-for-good>.
- This thread may also provide some helpful insight into the history of this module: <http://drupal.org/node/100971>.

Given that Viewfield enables views inside of views, and views in content items, with a variety of formats, there is no substitute to spending some time experimenting with Viewfield. Enjoy!

Recipe 29: Using the Node Reference module

Ingredients

Content module: <http://www.drupal.org/project/cck>
 Date module: <http://www.drupal.org/project/date>
 Date API module (installed with Date, above)
 Date Timezone module (installed with Date, above)
 Link module: <http://www.drupal.org/project/link>
 Node Reference module (installed with Content, above)
 media_hit content type
 press_release content type

The **Node Reference** module doesn't sound – at first – like it would have anything to do with Views. After all, it sounds like a module that would have something to do with referring to individual nodes. That's exactly what it does, enabling a field for referencing one node from another. In this example, we create a drop-down to link a News Hit to the Press Release that generated it. The press releases included in the Node Reference drop-down are determined by a View.

Enabling the modules

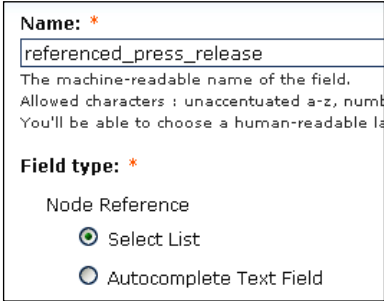
1. Go to `/admin/build/modules`. Enable the **Content** module if it isn't already enabled. Enable the **Node Reference**, **Date**, **Date Timezone**, **Date API**, and **Link** modules.

Creating the Media Hit and Press Release Content types

2. Create two content types, one named **media_hit** and one called **press_release** at `admin/content/types/add`.

Name	Media Hit	Press Release
Type	media_hit	press_release
Description	Each Media Hits will reference press releases	Press Release referenced by Media Hits

3. Add fields to the content type, as follows:

Adding fields to:	press_release	media_hit
URL to administer the fields	admin/content/types/media-hit/add_field (Use this URL for creating both fields below.)	admin/content/types/press-release/add_field (Use this URL for creating both fields below.)
Field names and field types	pr_release_date (Date - Select List) Deselect Hour and Minute in the Granularity settings	referenced_press_release (Node Reference - Select List)
		
	pr_release_body (Text - Text Field)	media_hit_url (Link - Text Fields for Title and URL)

After pressing **Create Field** there is no need to address the widget settings or data settings at the present time. We will pick up on this in Steps 6 – 9.

Adding Press Releases

- Go to `node/add/press-release` and populate the press releases with content. For instance, create Press Releases such as:
 - Food System Study Proves City Residents Lack Access to Fruits and Vegetables (July 2, 2009)
 - Garden of Eatin' Video Screening and Action Party (July 10, 2009)
 - The Farm Yard Project (July 13, 2009)

Creating drop-down Press Release View

5. Create a special view of press releases at `admin/build/views/add`. This view will be used to populate the Node Reference select list in the **Media Hit** content type.

Name	dropdown_press_releases
Description	Selection list used in Media Hit node reference field
Provide Page View	Checked
URL	<i>(No URL is necessary, since this view will be used to fill in a selection list.)</i>
View Type:	List view
Title:	Press Releases
Fields	Node: Title Date: pr_release_date (field_pr_release_date)
Filters	Node: Published Equals Yes Node: Type Is One Of Press Release
Sort Criteria	Node: Title Ascending (or pr_release_date, per your preference)

Remember to **Save** the view.

Modifying and populating the Media Hit Content type

6. Go to the following URL on your site.
`admin/content/types/media-hit/fields/field_referenced_press_release`

Here we are editing field options for the node reference in the **Media Hit** content type.

7. Change the label to **Related Press Release**.
8. Open the **Advanced - Nodes that can be referenced (View) fieldset**, and select the **dropdown_press_releases** view. Notice in this step that we also have the option just to select all of the Press Releases, via a checkbox, in the **Content types that can be referenced** section. This option would also populate the node reference but would only display the titles (also, the list would be limited to alphabetical order).

9. Press the **Save field settings** button.
10. Add a Media Hit at `node/add/media-hit`. Note that in the dropdown for the **Related Press Release**, the fields we selected in the **dropdown_press_releases** view are separated by a hyphen. The order matches the sort order we set in the view.

Submit Media Hit

Title: *

Related Press Release:

<none>
<none>
Dig This! Seminar - 06/23/2009
Food System Study Proves City Residents Lack Access to Fruits and Vegetables - 07/02/2009
Garden of Eatin' Video Screening and Action Party - 07/10/2009
The Farm Yard Project - 09/05/2009

11. Populate the rest of the **Media Hit**, including a media source URL, and click on **Submit**. Upon viewing the Media Hit you will see a link to the **Related Press Release**.

Related Press Release:
[Food System Study Proves City Residents Lack Access to Fruits and Vegetables](#)

Recipe notes

- Geoff Hankerson offers a terrific screencast on theming node reference fields, at:
<http://geoffhankerson.com/c/drupal/cck-nodereference-screencast.html>.

Recipe 30: Relating Content types with Reverse Node Reference

[Ingredients
Completed Recipe 29]

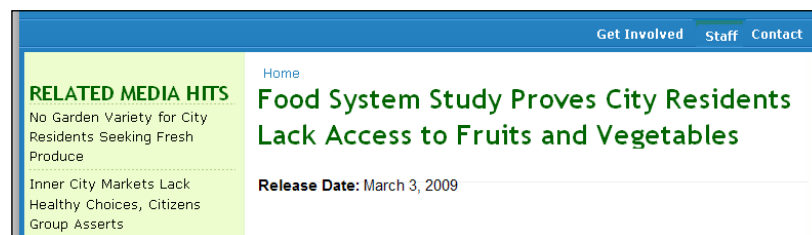
In the previous recipe, we linked a media hit to its associated press release. The link was made using a node reference selection list populated with a view of press releases and their release dates. In this recipe, we create a block view of media hits which displays associated press releases.

1. Complete *Recipe 29* including the creation of **Media Hits** and **Press Release** content types.
2. Create a view at `admin/build/views/add` as follows:

Name	related_media_hits
Description	Block view of media hits appearing beside related press releases
Provide Block	Checked
View Type:	List view
Title:	Related Media Hits
Nodes Per Block	6
[More] Link?	Checked
Fields	Node: Title - Option: As link
Arguments	Node Reference: Related Press Release (field_referenced_press_release)
Argument Handling Code	if (\$type == 'block' && arg(0) == 'node' && is_numeric(arg(1))) { \$nid = arg(1); \$args[] = \$nid; } return \$args;
Filters	Node: Published Equals Yes Node: Type Is One Of Media Hit
Sort Criteria	Node: Created Time Descending

Save the View.

3. Go to `/admin/build/blocks` and add the **related_media_hits** block to the Left sidebar.
4. Pull up a **Press Release** page which now displays a block of media hits that referenced it.



Recipe notes

- The key to this recipe is the use of arguments which we first discussed in Recipes 16 and 18. Let's say that our Press Release on the study of access to nutritious food is located at the following URL: `http://YOURSITE.com/node/233`. Note the value of the `arg()` functions based on the path components:

<code>arg(0)</code>	node
<code>arg(1)</code>	233

These path components will contain these values even if a URL alias is used. For instance:

`http://YOURSITE.com/food-system-study-proves-city-residents-lack-access-to-fruits-and-vegetables`

would share the same `arg` values even though `node/233` does not appear in the browser address bar. The argument field handling code assigns `arg(1)` to a node id variable named `$nid`. This variable is then assigned to the array of views arguments. Thus, each press release generates a different block of associated media hits.

- The drupal.org handbook includes some nice recipe pages that implement similar argument handling code to identify all nodes that are related by taxonomy or node author.

Add a 'related nodes' block that links to a taxonomy-based view	http://drupal.org/node/65375
---	---

Views block that pulls content based on a page's node author (UID)	http://drupal.org/node/174281
--	---

- Note that argument handling code does not use opening or closing php tags. These tags are required, however, in new block code and **Header**, **Footer**, and **Empty text** in views (also, the **Input format** for code must be set to **PHP Code**, an option that is not available for Argument code).

Recipe 31: Embedding a View in a Theme template

Ingredients

Content module: <http://drupal.org/project/cck>
 Node reference field, included in CCK module
 Automatic Nodetitles: http://drupal.org/project/auto_nodetitle
 Token module: <http://drupal.org/project/token>
 Imagefield module: <http://drupal.org/project/imagefield>
 Artist Content type
 Poster Content type and View

This recipe performs a similar function as the Recipes 28 and 30, but takes a different approach. Using a template file, and a small mix of modules and views functions, we will embed all posters produced by an artist under the artist's description.

1. Create a content type named **artist** to include the following text fields.

Field Name	Field Type	Label
artist_first_name	Text	First Name
artist_last_name	Text	Last Name
artist_nationality	Text	Nationality

2. Edit the content type at `admin/content/types/artist` and select **Automatically generate the title and hide the title field**.

Automatic title generation

☐ Disabled

☒ Automatically generate the title and hide the title field

☐ Automatically generate the title if the title field is left empty

Pattern for the title:

[field_artist_first_name-formatted] [field_artist_last_name-formatted]

Leave blank for using the per default generated title. Otherwise this string will be used as title. Use the syntax [token] if you want to insert a replacement pattern.

Replacement patterns

- To determine the bracketed pattern names, click on the **Replacement patterns** fieldset and scroll down to the CCK replacement patterns. These bits of text are made available by the **Token** module.

CCK text tokens	
[field_artist_first_name-raw]	Raw, unfiltered text.
[field_artist_first_name-formatted]	Formatted and filtered text.
[field_artist_last_name-raw]	Raw, unfiltered text.
[field_artist_last_name-formatted]	Formatted and filtered text.

- Add **[field_artist_first_name-formatted]**, and **[field_artist_last_name-formatted]** to the **Pattern for the title**. Make sure to separate the two tokens with a space.
- Save the content type.
- Create a **dropdown_artists** view at `admin/build/views/add`, to populate the poster node reference (this is very similar to the **dropdown_press_releases** view in *Recipe 28*).

Name	dropdown_artists
Description	Selection list used in Artist node reference field
Provide Page View	Checked
URL	<i>(No URL is necessary since this view will be used to fill in a selection list.)</i>
View Type:	List view
Title:	Artists
Fields	Node: Title <i>(This field will be populated via auto nodetitle)</i>
Filters	Node: Published Equals Yes Node: Type Is One Of Artist
Sort Criteria	Text: Last name (field_artist_last_name) - Ascending Text: First name (field_artist_first_name)

- Create a **Poster** content type, with the following fields:

Field Name	Field Type	Label
poster_image	Image	Poster Image
poster_artist	Node Reference	Artist

When configuring the node reference field at: `admin/content/types/poster/fields/field_artist`, be sure to select the **dropdown_artists** view to populate the node reference field. This is done in the **Advanced - Nodes that can be referenced (View)** fieldset.

Of course, you may add other fields as well to describe the poster, such as date, and language.

8. Populate the **Artist** and **Poster** content types with sample data.
9. Create a simple view of posters with the following settings:

Name	artist_posters_view
Description	List of posters for each artist, for inclusion in node-artist.tpl.php
Provide Page View	Checked
URL	<i>(No URL is necessary.)</i>
View Type	List View
Title	<i>Leave blank as this will be placed by the template.</i>
Use Pager	Checked
Nodes Per Page	10
Fields	Node Title - Option: As link
	Image: Poster image (field_poster_image)
Arguments	Node Reference: Artist (field_poster_artist)
Filters	Node: Published Equals Yes
Sort Criteria	Text: Last name (field_artist_last_name) - Descending

10. **Save** the view and review it at `admin/build/views/artist_posters_view`. This will display all the posters. You may also filter by artist, by appending the artist ID to the URL, for instance, `admin/build/views/artist_posters_view/68` (be sure to use an ID that belongs to an artist node).
11. Review the handbook page at `http://drupal.org/node/124446`.
12. Login to your host. Go to your theme directory which will be often located at `<DRUPALROOT>/sites/all/themes`. Copy **node.tpl.php** to **node-artist.tpl.php**.
13. Copy the following PHP code from the handbook page to an appropriate place in the file (in the Zen theme, this will be just above the closing `</div>` tags. In many other themes it will be just under the div containing the `$content` variable).

```
<?php
// load the context-node's 'metadata'
global $current_view;
// * define the context-node's NID as the argument
$current_view->args[0]=$node->nid;
// * select the name of the view to embed as $view1
$view1 = views_get_view('issue_view'); // Line 7
// * define this section for CSS
print '<div class="issue_view">'; // Line 9
/* display a subtitle for the view section
print '<h2>' . t($node->title . ' Issues') . '</h2>'; //Line 11
// send $args to the View's Argument Handler and display $view1 in
the context-node
print (views_build_view('embed', $view1, $current_view->args,
false, false)); // Line 13
print '</div>';
?>
```

We'll make three modifications in this code:

Line 7: `$view1 = views_get_view('artist_posters_view');`

This gets the view we created above. The view includes the title and image for each poster.

Line 9: `print '<div class="artist-posters-view">';`



It is a convention to use hyphen rather than underscores in CSS class names. This convention comes from a time when browsers could not recognize the underscore character in a class name.

Line 11. `print '<h3>' . t($node->title . ' Posters') . '</h3>';`

This line prints the artist's name (the automatically generated title of the artist field) enclosed in an `<h3>` tag.

The key to this code snippet is line 13 which builds an embedded view, limiting the results by the artist.

14. View an Artist node on the site, for instance: `http://YOURSITE.com/node/68`. The artist's posters appear below the artist description.



Recipe notes

- Documentation for the views functions used in the snippet may be found at:
views_get_view
http://api.freestylesystems.co.uk/api/function/views_get_view/5
views_build_view
http://api.freestylesystems.co.uk/api/function/views_build_view/5
- The `views_build_view` function is unique to Views 1. Similar functionality can be found in Views 2 called `views_embed_view`.
- One reason for choosing this theme-based technique over the one in the previous recipe is that the Drupal site administrator needn't have any interaction with decisions about Views. The `artist_posters_view` appears automatically, whenever an artist node is viewed. Conversely, an administrator who does not want to deal with (or does not have access to) theme files may choose an approach such as *Recipe 29*.
- More information can be found on embedded views at:
<http://drupal.org/node/48816>.

Recipe 32: Displaying mini-pie charts

Ingredients

CCK module: <http://drupal.org/project/cck>
Number field module, included in CCK module
CCK Computed Field module http://drupal.org/project/computed_field
Devel module: <http://drupal.org/project/devel>
Regenerate module: <http://drupal.org/node/195013#comment-1168380>
Unzip utility such as WinZip, WinRAR, or unzip
sIFR module <http://drupal.org/project/sifr>
sIFR <http://www.mikeindustries.com/sifr/>
Pie chart for maps sIFR font, <http://www.drupalviews.org/node/1>

This view uses a special font to help visualize a percentage.



The recipe implements the "Pie Charts for Maps" font originally created by Curtis Clark as a truetype font. I converted the font into a Flash-based format known as **Scalable Inman Flash Replacement (sIFR)**. sIFR fonts can be viewed in browsers that have both Flash and JavaScript enabled. sIFR provides the best of the Flash and HTML worlds: we get the creative font choices afforded by Flash, while also keeping the accessibility of HTML. An accessible page ensures that blind web users, as well as search engines, will be able to access the content.

If the user does not have JavaScript or Flash enabled, the original HTML text with CSS will be displayed. This recipe will break for people who have enabled Flash blockers for their browser. The vast majority of users, however, have Flash and JavaScript turned on.

Setting up the Pie Chart Content type and View

1. Install and enable the first five modules in the ingredients list. Note that the Regenerate module is not located in the regular Drupal contributed module repositior: it is a small module that a user uploaded to the module's issue queue. This module overcomes a quirk in the **Computed Field** module that changes in the computed field require each node to be re-saved. This module does a bulk save of each node in a selected content type—a huge time-saver and annoyance-reducer.
2. Create a new content type called **piechart** at `admin/content/types/add`.

Name	Pie Chart
Type	piechart
Description	Content type for use with Pie4Map.swf
Default Options	Check Published Uncheck Promoted to the frontpage

3. Add two fields to the piechart content type at **Administer | Content management | Content types | Pie Chart | Add Field** (admin/content/types/piechart/add_field):

Field name	Field type	Custom Values
percentage	Decimal - Text field	Label: Percentage Minimum Value: 0 Maximum Value: 100 Suffix: %
computed	Computed - Computed	Label: Computed Computed field: <i>See code block below.</i> Display this field: Checked Display format: \$display = \$node_field_item['value']; Store using the database settings below: Checked Data Type: varchar Data Length: 1

Enter the following **Computed field** code: (*note that the PHP tags are not included*)

```
$percent = $node->field_percentage[0]['value'];  
$percent = 5 * round($percent/5);  
$fontvalue = array(  
  '0' => 'A',  
  '5' => 'B',  
  '10' => '1',  
  '15' => 'D',  
  '20' => '2',  
  '25' => 'F',  
  '30' => '3',  
  '35' => 'H',  
  '40' => '4',  
  '45' => 'J',  
  '50' => '5',  
  '55' => 'L',  
  '60' => '6',  
  '65' => 'N',  
  '70' => '7',  
  '75' => 'P',  
  '80' => '8',  
  '85' => 'R',
```

```

    '90' => '9',
    '95' => 'T',
    '100' => 'U',
  );
  $node_field[0]['value']=$fontvalue[$percent];

```

This code rounds the value in the percentage field to the nearest 5 and maps that value to a character displaying the appropriate pie chart in the Pie4Map font.

4. Populate the **Pie Chart** content type. Note that the **Computed Field** does not show up on the edit page. It is being calculated "behind the scenes" and will appear in Views.

Submit Pie Charts

Title: *

percentage:

 %

5. Create a simple view of **piecharts**, displaying the **Name**, **Percentage**, and **Computed** fields.

Name	piecharts
Description	Pie Chart
Provide Page View	Checked
URL	piecharts
View Type:	Table View
Title:	Pie Charts
Fields	Node: Title Decimal: Percentage (field_percentage) Computed: Computed (field_computed)
Filters	Node: Published Equals Yes Node: Type Is One Of Pie Chart
Sort Criteria	Node: Title Ascending

6. **Save** the view. At this time, the view looks like this:

Adam	50%	5
David	90%	9
Earl	100%	0
John	33%	H
Lucretia	66%	N
Maggie	10%	1

The value for **Lucretia**, for instance, is rounded from 66 down to 65, and then assigned the value of **N** (the letter assigned the pie chart matching the value 65%).

While at this URL, note that the CSS rule for the computed table cell is as follows:

.view-field-node-data-field-computed-field-computed-value

This may be determined via the Firebug, or by simply viewing the source code of the view page.

Install and configure sIFR

7. Install and enable the sIFR module <http://drupal.org/project/sifr>.
8. Create a sifr subfolder of the sifr folder at `sites/all/modules/sifr/sifr`.
9. Download sIFR at the bottom of the page at <http://www.mikeindustries.com/blog/sifr/>.
10. Copy the following files of that archive into `sites/all/modules/sifr/sifr`:
 - ❏ `sifr.js`
 - ❏ `sifr-addons.js`
 - ❏ `sIFR-print.css`
 - ❏ `sIFR-screen.css`



Steps 7–10 may be accomplished at the command line with the following steps (*be sure to check for current versions in the wget statements, and substitute the actual directory path for <DRUPALROOT>*).

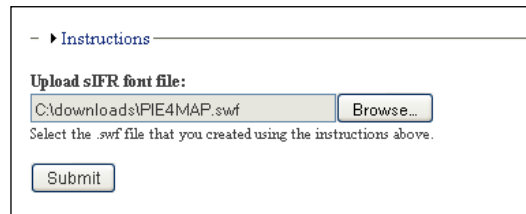
```
cd <DRUPALROOT>/sites/all/modules
wget http://ftp.drupal.org/files/projects/sifr-5.x-1.3.tar.gz
```

```

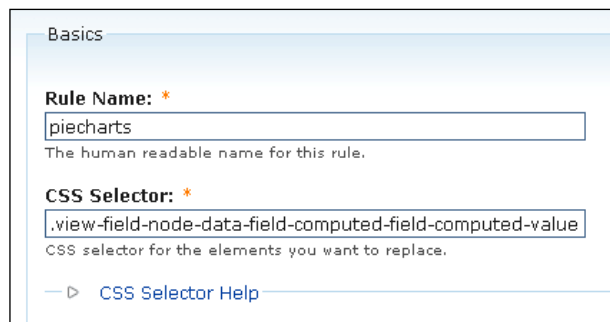
tar xvf sifr-5.x-1.3.tar.gz
cd sifr/
mkdir sifr
cd <DRUPALROOT>
mkdir downloads
cd downloads
wget http://www.mikeindustries.com/blog/files/sifr/2.0/
sIFR-2.0.7.zip
unzip sIFR-2.0.7.zip
cp *.js <DRUPALROOT>/sites/all/modules/sifr/sifr
cp s*.css !$

```

11. Download the pie chart for maps sIFR font from <http://www.drupalviews.org/node/1>.
You could also download the sIFR font in the code bundle available at: http://www.packtpub.com/files/code/6965_Code.zip
12. Upload your pie4map.swf to your site, from admin/settings/sifr/manage.



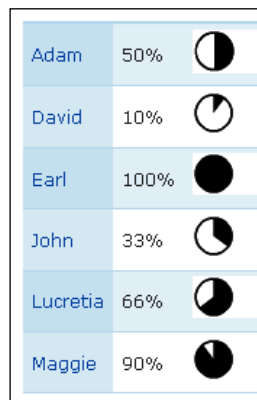
13. Click on the **Add Rule** tabs. Enter a **Rule Name** and the **CSS Selector** for the computed field column.



14. Select the Pie Chart font to apply to this rule.



15. Return to the View.



Recipe notes

- This example uses the Drupal sIFR module. Similar sIFR functionality exists in the Render module found at <http://drupal.org/project/render>.
- If you find the field behaving unexpectedly after you modify the computed field and regenerate the nodes, try clearing the views cache.

Summary

In this chapter we resolved a body field quirk using a relatively new module to copy the body field to a CCK text field. We looked at *formatters*, which provide module-specific options for displaying fields in both CCK and Views (*Appendix B* contains a lengthy list of formatters available by module). We cloned a view as a starting point for a new view, in this case, to create a blogroll. We also looked at a variety of ways to associate views with related content including viewfield, node reference, and building views from within a template. In the last recipe, we developed a data visualization view offering a foundation for additional creative approaches both to calculated fields and Flash fonts.

4

Dates and Calendars

All of the recipes in this chapter require CCK and the date field (also Views, and Views UI), so we won't take the space to list them in every ingredient list. In this chapter, we create a wide variety of views, mostly from a single workshop content type. Several of the views implement style plug-ins, which appear as new options in the **Views Type** drop-down. If you are still working on the 1.x versions of Date, be sure to check out *Recipe 44, Upgrade Date and Calendar Modules*.

Recipe 33: Understanding Date formats

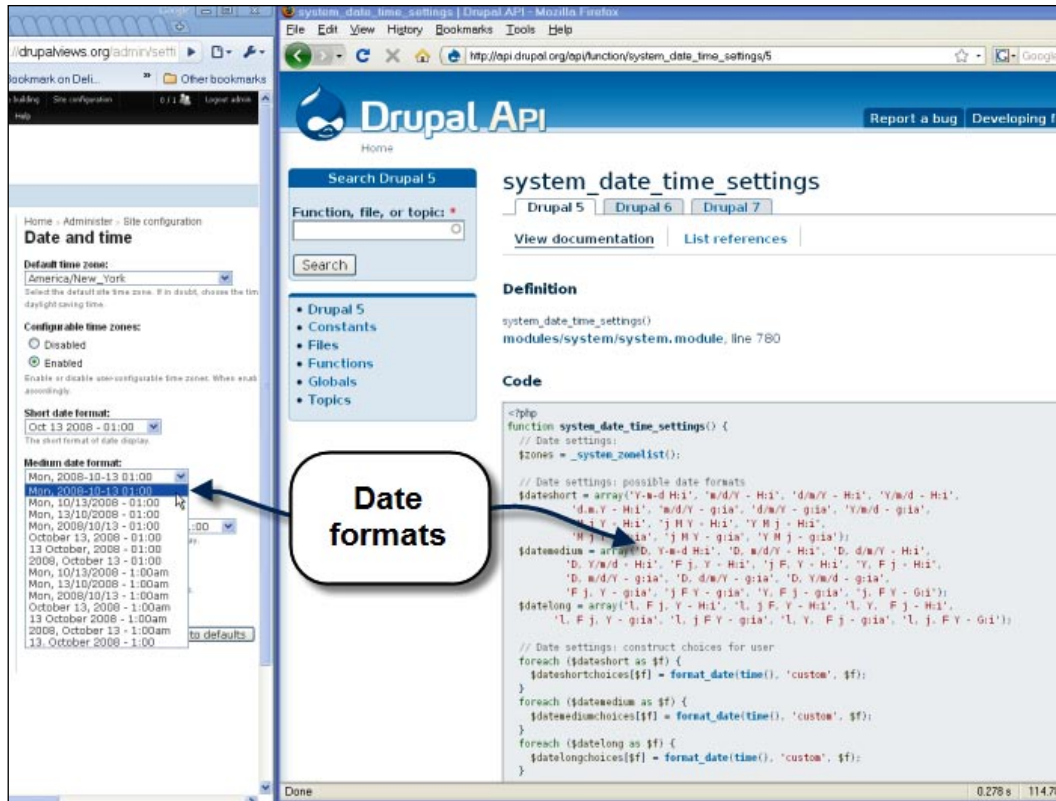
Drupal dates are typically stored in one of two ways. Core Drupal dates – including **Node: Created Time**, and **Node: Updated Time** – are stored as Unix timestamps. Contributed module date fields can be stored as either a timestamp or a format known as ISO. Neither style is particularly friendly to human readers, so both field types are usually formatted before users see them. This recipe offers a tour of places in Drupal where dates can be formatted and information on how to customize the formats.



What's that Lucky Day?

The Unix timestamp **1234567890** fell on Friday the 13th, in February, 2009. This timestamp marks 1,234,567,890 seconds since January 1, 1970. The same date/time combination would be stored in a date field in ISO format as **2009-02-13T23:31:30+00:00**. ISO is an abbreviation for the International Organization for Standardization.

1. Opening the browser windows side-by-side will help you understand date formatting. In the left window, open YOURSITE.com/admin/settings/date-time to view the settings page for date and time. In the right window, open the API page of code that defines these system date time settings at http://api.drupal.org/api/function/system_date_time_settings/5. Compare each item in the **\$datemedium** array, for instance, with the associated **Medium date format** drop-down.



Below is the list of codes for many commonly used date and time formats. A more comprehensive list appears at <http://us.php.net/date>.

a - am/pm

D - Day, Mon through Sun

d - Date, 01 to 31 (with leading zeroes)

F - Month, January through December (mnemonic, F = Full name)

g - Hours, 1 through 12

H - Hours, 00 through 23

- i – Minutes, 00 to 59
- j – Date, 1 to 31 (No leading zeroes)
- l – Sunday through Saturday
- m – Month, 01 through 12
- M – Month, Jan through Dec
- s – Seconds, 00 through 59 (with leading zeroes)
- S – Month Suffix, st, nd, rd, or th. Works well with j
- Y – Year, Examples: 1999 or 2011

2. Explore Drupal places where these codes may be used. The first four locations in the table below are available in the Drupal administrative interface. The last three involve editing files on the server – these edits are completely optional.

Location	Details
CCK field setup	Custom Input formats admin/content/types/story/add_field

The screenshot shows a dialog box with two sections: 'Date' and 'Datestamp'. Each section has three radio button options: 'Select List', 'Text Field with custom input format' (which is selected), and 'Text Field with jquery pop-up calendar'.

After the field widget is specified

admin/content/types/<CONTENTTYPE>/fields/
field_<FIELDNAME>

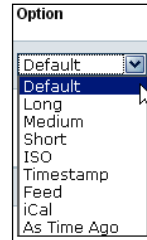
Near the top of the page:

The screenshot shows a dialog titled 'Customize Default Value'. It has an 'Input format:' dropdown menu showing '2008-10-26 20:52:17'. Below it is a text area for '*Custom input format:' which is empty. A note at the bottom states: 'The custom format, if provided, will override the input format selected above. See more about custom date formats below.'

Near the bottom of the page:

The screenshot shows a dialog titled 'Default Display'. It has a 'Date display:' dropdown menu showing '10/26/2008 - 8:49pm'. Below it is a text area for '*Custom display format:' which is empty.

Location	Details
Formatting Fields in Views	admin/build/views/<VIEWNAME>/edit CCK Date fields are set via the Option drop-down in the Fields fieldset .



Custom date formats for core fields, such as **Node: Created Time** are set via handler and option form elements.

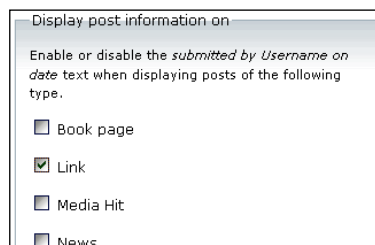


Default Date and Time settings

admin/settings/date-time
Set the default time zone, Short, Medium, and Long date formats, and the first day of the week.

Post Settings

This may be one of the harder-to-find settings in Drupal. It enables the **Post Settings** to be turned-off for specified content types. (An example of a post setting would be: *Submitted by admin on Sun, 10/12/2008 - 4:55pm.*) The setting is found on the right-hand side of this URL: admin/build/themes/settings
Use the following mouse click trail to get to this URL:
Administer | Site Building | Themes | Configure
(Click on the **Configure** tab at the top of the page. If you click on the Configure link in the **Operations** column, you will still need to click the **Configure** tab at the top to get to the global settings.)



Location	Details
Variable overrides in settings.php	You may override variables at the bottom of the sites/default/settings.php file. Remove the appropriate pound signs to enable the \$conf array and add the setting as shown below. Note that this is a quick way to modify the post settings format, which draws from the medium date variable. <pre>\$conf = array(# 'site_name' => 'My Drupal site', # 'theme_default' => 'minnelli', # 'anonymous' => 'Visitor', 'date_format_medium' => 'l F d, Y');</pre>
*.tpl.php files	Examples: - node-story.tpl.php <pre><?php print format_date(\$node ->created, 'custom', 'F Y'); ?></pre> - comment.tpl.php <pre><?php echo t('On ') . format_ date(\$comment->timestamp, 'custom' , 'F jS, Y'); ?> <?php echo theme('username', \$comment) . t(' says:'); ?></pre>
template.php	Redefine \$variables['submitted'] Example from blommor01 theme: <pre>\$vars['submitted'] = t('!user - <abbr class="created" title="!microdate">!date</abbr>', array('!user' => theme('username', \$vars['node']), '!date' => format_date(\$vars['node'] ->created), '!microdate' => format_ date(\$vars['node']-> created, 'custom', "Y-m-d\TH:i:sO")));</pre>

Recipe notes

Note that when using the PHP date codes, additional characters may be added, including commas, spaces, and letters. In the `template.php` example, a backslash was used to show that the letter 'T' will be printed, rather than the formatted return values. Below are more examples of added characters:

```
F j, Y, g:i a    // August 27, 2010, 5:16 pm
m.d.y          // 08.27.10
```

You may occasionally find that an online date converter comes in handy.

- <http://www.timestampconverterer.com/>
(this URL includes the word "converter" followed by another "er").
- <http://www.coryking.com/date-converter.php>

Recipe 34: Block of upcoming workshops

Ingredients

CKK module: <http://drupal.org/project/cck>
Date module: <http://drupal.org/project/date>
Enable Date, Date API, Date Popup, and Date Timezone modules
Node Go To module: <http://drupal.org/project/nodegoto>

This recipe is the foundation for many others in the book, including most the ones remaining in this chapter.

The **Node Go To** module is a little-known treasure that makes it easy to enter content. This recipe also makes use of the **now** option, available in the **Date** module.

1. Install and enable all of the modules in the ingredients.
2. Go to `admin/settings/date-time` and set the default time zone.
3. Go to `admin/content/types/add` and create a content type named **Workshop**, of type **workshop**. As usual, remove the **Body** field in the **Submission form** fieldset. We'll create a **workshop_description** field instead.

- Go to `admin/content/types/workshop/add_field` and add a field named **workshop_date**. We have several field types to choose from, however, I recommend the **Datestamp** format for greater flexibility. For instance, in *Recipe 80*, we will be using Drupal's `format_date()` function to create graphical date badges. This function requires a Unix timestamp as the first parameter (the Drupal Date module uses the term **Datestamp** for fields stored in Unix timestamp format). The **Text Field with jquery pop-up calendar** offers a friendly way of entering dates.

The screenshot shows a form titled 'Date' and 'Datestamp'. Under 'Date', there are three radio button options: 'Select List', 'Text Field with custom input format', and 'Text Field with jquery pop-up calendar'. Under 'Datestamp', there are also three radio button options: 'Select List', 'Text Field with custom input format', and 'Text Field with jquery pop-up calendar'. The 'Text Field with jquery pop-up calendar' option under 'Datestamp' is selected. At the bottom of the form is a 'Create field' button.

- Press the **Create field** button. All the settings at `admin/content/types/workshop/fields/field_workshop_date` may be left as default, or edited, as per your preference. Click the **Save field settings** button.
- Go to `admin/content/types/workshop/add_field`, and add a text field named **workshop_description**. Click the **Create field** button.
- Set the number of rows to **4** and set the **Text processing** to **Filtered text (user selects input format)** at `admin/content/types/workshop/fields/field_workshop_description`. Click the **Save field settings** button.
- The **Node Go To** module redirects the browser to a specified URL after creating, updating, or deleting a node. Configure the module at `admin/settings/nodegoto`. Open the fieldset labeled **Set the path to redirect for, when INSERTING**. Enter **node/add/workshop** as the path as shown:

The screenshot shows a form titled 'Insert the path for Workshop:'. Below the title is a text input field containing the text 'node/add/workshop'.

- Create several workshops, at `node/add/workshop`. As a result of the configured **Node Go To** module, after each node is submitted, the entry form for another workshop appears.

10. Create a new view, at admin/build/views.

Field	Value
Name	workshops
Description	Page and block of workshops
Provide Page View	Checked
URL	workshops
View Type	Full Nodes
Title	Upcoming Workshops
Use Pager	Checked
Nodes per Page	10
Provide Block	Checked
View Type	List View
Title	Upcoming Workshops
Nodes Per Block	6
[More] Link?	Checked
Fields (use default handlers and options.)	Node: Title Datestamp: workshop_date (field_workshop_date)
Filters	Node: Published Equals Yes Node: Type Is One Of Workshop Datestamp: workshop_date - Date (field_workshop_date) Set the Operator to greater than or equal to . Enter the word now in the Option field. Thus we are filtering to workshops that will be held in the future.
Sort	workshop_date (field_workshop_date) Ascending

11. **Save** the view.

12. Go to admin/build/block and add the **workshops** block to the right sidebar.
Enjoy the view.

Upcoming Workshops

- Worm Composting: Turning Your Kitchen Scraps Into Compost with an Indoor Box of Worms
Jun 8 2009
- Video: Percy Schmeiser, the Canola Farmer Who Fought Monsanto
Jun 15 2009
- Sprouting, From Alfalfa and Beans to Sunflower Greens
Jun 22 2009
- Sea Supper Supreme
Jun 29 2009

13. We will add one more set of steps in preparation for Recipes 35 – 41. Clone this view six times at `admin/build/views/workshops/clone`. Rename each view as follows:

Name	Description	URL
exposed_filters	Exposed Filters	exposed-filters
summary	Summary	summary
browse_workshops	Browse Workshops	browse-workshops
timeline	Timeline	timeline
views_popup	Views Popup	views-popup

For each view you may also uncheck the **Provide Block View** checkbox and, **Save** the view.

Recipe notes

- The **Submit More** module offers similar functionality to **Node Go To**. http://drupal.org/project/submit_more.
- The **GCal Events** module http://drupal.org/project/gcal_events offers a handy way of creating blocks of upcoming events. Events are drawn from Google Calendar.
- If you are still using Date 1.x, be sure to install the JavaScript Tools module and enable the JSTools Calendar module. This will give you a pop-up calendar selection tool similar to the one available in Date 2.x. You may also implement *Recipe 44* to upgrade the Date module.

Recipe 35: Exposed Filters

Ingredients

Completed Recipe 34, exposed_filters view

We finally turn to the **Exposed Filters** fieldset, which exposes one or more selection lists to users, enabling them to choose how to filter the view (until now, all of our filters have been applied by the web site administrator, rather than by end users). In this recipe, we enable users to filter by the date field. We will explore the **Views Date Range Filter** module which offers additional options, in the recipe that follows.

1. Edit the **exposed_filters** view, at `admin/build/views/exposed_filters/edit`.
2. Go to the **Filters** fieldset and click on the **Expose** button for the **Datestamp: workshop_date** field.



The delete icon disappears from the **Ops** column for that row. However, the field now appears in the **Exposed Filters** fieldset. If you want to delete a field that has been exposed, you must first delete it from the **Exposed Filters**, before it is available to delete in the **Filters** fieldset.

3. Open the **Exposed Filters** fieldset. Review how it looks, but leave the default settings intact. (Note that there is no **Add Filter** drop-down available. **Exposed Filters** field are added and exposed via the **Filters** fieldset, as in step 2, above.)
4. **Save** your view and go to `http://YOURSITE.com/exposed-filters`. A new selection list is available.

greater than or equal to
2009-06-21 00:00
Format: 2008-10-14 13:39

Submit

Sprouting, From Alfalfa and Beans to Sunflower Greens

June 22, 2009



Learn to grow fresh greens, all year long. (If you've never had sunflower greens before, you're in for a delightful treat.)

Recipe notes

- **Exposed Filters** offer powerful search capabilities to end users.
- Explore the `date_views_filter_handler` in `date_views.inc` to see the code behind date filters. The code is complex, as is typical of date functions, but well-commented, especially in the **Date 5.x-2.x** version.

Recipe 36: Views Date Range Filter module

Ingredients

Completed Recipe 35

Views Date Range Filter module: <http://drupal.org/project/daterange>

1. Install and enable the **Views Date Range Filter** module.
2. Edit the **exposed_filters** view at `admin/build/views/exposed_filters/edit`. Go to the **Exposed Filters** fieldset and click on the delete icon to remove the **Datestamp** filter.
3. Click on the **Save and edit** button (always a good idea after working with the **Exposed Filters** fieldset).
4. Go to the **Filters** fieldset and delete the **Datestamp** filter there too.
5. Add the **Date Range Filter** and press the **Expose** button.
6. **Save** the view and test the various date range options. Compare with the date filter options from the previous recipe.

The selection list for the **Views Date Range Filter**

Compare with the drop-down list from *Recipe 35*

In addition to all the preset ranges (**within the past week** and **in the next month**), the **Date Range** module lets you choose events occurring between two dates (**Custom date range**) It even allows you to exclude a range of dates (**Outside date range**) and to specify dates such as **today**, **yesterday**, **today-7**, **today-2M**, or **today-1Y**.

Recipe notes

- There are some other subtle differences to be found when filtering by date versus filtering by a Date Range. The **Views Date Range Filter** module allows placement of values such as **today+1** or **today-1Y** in the **Value** text box of the **Filters** fieldset. By contrast, **Date 2.x** allows values such as **now** and **now +1 day**. *Spacing* and *capitalization* are important. Copy and paste from the helpful notes under the field, or from the `README.txt` file to ensure the appropriate format.

Recipe 37: Exposed Filter settings

[**Ingredients**
Completed Recipes 34 and 35]

The **Exposed Filters** fieldset provides four options, as shown:

Optional	Filter settings Default	Force Single	Lock Operator
☐	☐	☐	☐

The handbook page for filters (<http://drupal.org/node/54457>) offers helpful documentation on these options. After studying that page, the best way to get to understand these filter options is simply to try them. While testing the options keep an eye on both the **Filters** and **Exposed Filters** fieldsets (two of the options: **Filter settings Default** and **Lock Operator** correspond directly to settings in the **Filters** fieldset).

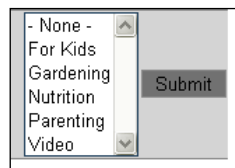
1. Create a vocabulary for workshop topics at `admin/content/taxonomy/add/vocabulary`.

Vocabulary name	Topic
Description	Topics for workshops
Help Text	Select one or more topics
Types	Workshop
Hierarchy	Disabled
Check	Multiple Select and Required.

2. Add terms such as **Nutrition**, **For Kids**, and **Gardening** to the **Topic** vocabulary at `admin/content/taxonomy`.
3. Edit the existing workshops nodes and assign appropriate terms. It may be helpful to use the view to identify the workshops. Open the **exposed_filters** view and instead of choosing **Edit** (which will edit the view), click on each title link. Now choose **Edit** to edit that particular node. For the sprouting workshop, assign to it the topics such as **Nutrition** and **Gardening**.
4. Edit the view at `exposed_filters/edit`. Add the **Taxonomy: Terms for Topic** taxonomy to the **Filters** fieldset and click **Expose**.
5. Experiment with the **Exposed Filters** options. Click the checkbox for each of the four **Exposed Filters** options, save the view, and examine the results. Your results may appear slightly different than the images below depending on whether your taxonomy is set to be required and depending on whether several options are checked at once.

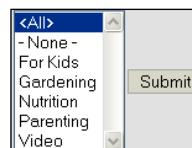
Optional: Unchecked

Does not include an **All** selection and no workshops are displayed by default.



Optional: Checked

Includes an **All** selection and all workshops are displayed.
(The **None** option only appears if the taxonomy is not required.)



Filter settings Default: Unchecked

All of the workshops are displayed.

Filter settings Default: Checked

Only the videos are shown, by default, at the exposed_filters URL. The **Video** selection was set in the **Filters** fieldset.

Force Single: Unchecked

Force Single: Checked

Lock Operator: Unchecked

Lock Operator: Checked

The **Is One Of** operator from the **Filters** fieldset is locked in for use by Exposed Filters. Thus, the option does not appear for end users.

Recipe notes

- If you lock an operator, consider using the Field's **Label** to give some information on how to use the drop-down values.

Recipe 38: Summary Views

[Ingredients
Completed Recipe 34, Summary View]

Views offers a wonderful feature, called **Summary Views**, available in the **Argument** fieldset. It creates a hyperlinked list of topics with a count of items.

1. Edit the **summary** view, created in *Recipe 34*.
2. Add the **Datstamp: Workshop Date** argument, with the following settings:

Default	Summary, sorted ascending
Title	Workshops in %1
Option	summarize by month
Wildcard	all

Argument Type	Default	Title	Option	Wildcard	Wildcard Sub
Date: workshop_date (field_workshop_date)	Summary, sorted ascending ▼	Workshops in %1	summarize by month ▼	all	

3. Enjoy the View.

Workshops

- [June, 2009](#) (4)
- [July, 2009](#) (4)
- [August, 2009](#) (5)
- [September, 2009](#) (4)
- [October, 2009](#) (4)

4. Click on a month link. Notice that the month is included as an argument in the URL. The title of the page substitutes the month for the %1 (%1 was also used in *Recipe 16*).

Workshops in September, 2009

5. Try out the wildcard at <http://YOURSITE.com/summary/all>
All the workshops are displayed, using the pager count of 10.
6. We would like the view to display up to 10 workshops per page but prefer to display the next *12 months* in the Summary View – so we will add argument handling code to the view. Edit the view at `summary/edit`. In the **Arguments** fieldset open the **Argument Handling Code** and add the following:

```
if (!$args[0]) {
    $limit = 12;
}
```

Thus, if the argument in the **workshop_summary** view is not set (for instance, at the URL <http://YOURSITE.com/summary>, as opposed to <http://YOURSITE.com/summary/2010-02> or <http://YOURSITE.com/summary/all>), then the page limit will be 12 rather than the usual limit of 10 items that were set in the **Page** fieldset.

You will now see all 12 months in the summary – if you actually have 12 months of upcoming workshops (review *Recipe 18* in Chapter 2 for more detail on the \$args array).

7. (Steps 7–9 are optional.) If you would rather have the view summary display workshops by topic, edit the view at `summary/edit`.
8. Delete the argument and the argument handling code. Replace it with **Taxonomy: Term Name**.

Default	Summary, sorted ascending
Title	Workshops - %1
Option	(leave blank, or set to 0)
Wildcard	all

Argument Type	Default	Title	Option	Wildcard	Wildcard Sub
Taxonomy:					
Term	Summary, sorted ascending	Workshops		all	
Name					

9. **Save** the view and open it in your browser.

[Home](#)

Workshops

View successfully saved.

- [Bodywork](#) (5)
- [Cooking](#) (5)
- [Crafts](#) (2)
- [For Kids](#) (1)
- [Gardening](#) (2)
- [Parenting](#) (2)
- [Potluck](#) (2)
- [Video](#) (2)

Recipe notes

- The Drupal handbook contains a nice view for creating a monthly archive of content. Even nicer, it is something that can be easily imported into your site. Go to <http://drupal.org/node/52037>, copy the **Import this view** code block and paste it into `admin/build/views/import`. Save the view and place the archive block into a region at `admin/build/blocks`. The archive view is similar to the one in Step 2 above, but it is based on the node created time rather than a CCK date field. It also sorts in descending order (with the most recent month at the top) and includes all site content, by default.
- The Views wildcard is commonly set to **all**. Of course, if **Nodes per Page** is set to **10**, it really only shows 10 nodes at a time and not all of them. It may be preferable to set the wildcard to the word **browse**, making the URL: `YOURSITE.com/<VIEW-NAME>/browse`.

Recipe 39: Date Browser

[**Ingredients**]
Completed Recipe 34

Views, by default, come with four View types: **Full Nodes**, **Teaser List**, **Table View**, and **List View**. The **View Types** selection list is populated by **Views style plugins** coded in modules. These **style plugins** add options to the **View Types** drop-down in both the **Page** and **Block** fieldsets. Style plugins can provide dramatic alterations in node formats. We'll have a chance to look at a date browser, a timeline, a views popup, and a calendar, in Recipes 39 - 42. A comprehensive list of Drupal 5 style plugins is included in *Appendix C*.

1. Edit the **browse_workshops** view created in *Recipe 34*. As usual, you may visit either `browse_workshops/edit` (this URL uses the general theme) or `admin/build/views/browse_workshops/edit` (this URL uses the administrative theme).
2. Set the **Page View Type** to **Date: Date Browser** and **Save** the view. A message is displayed:

A date argument must be added to a Date Browser view.

3. Add the **Datestamp: field_workshop_date** field to the **Arguments** fieldset and **Save** the view. Another message is displayed:

Date Browser arguments must be set to 'Display All Values'.

The **Date** module is kindly guiding us through the appropriate settings for the Date Browser view type.

- Go back to the **Arguments** fieldset and set the **Default** to **Display All Values**. While there, also set the **Option** to **summarize by month**.

Argument Type	Default	Title	Option
Datestamp: workshop_date (field_workshop_date)	Display All Values		summarize by month

- Save the view and browse to the **browse_workshops** page. The first page displays the current month, with a pager. Additional links are displayed for **next month** and **previous month**.

Workshops

[< prev month](#)
[Month of September 2009](#)
[next month >](#)

Labor Day Hike and Picnic

Potluck
September 7, 2009

Meet at 11:30 at the Shady Park Pavilion. Bring your favorite healthy dish, recipe card, comfortable hiking shoes, musical instruments, swimsuit, and game items (volleyball, etc.)

Hikes (4 or 7 miles) are scheduled to leave at 2pm. We have the Pavilion until 7 pm.

Rawsome Food Prep with Keisha

Cooking
September 14, 2009



Keisha shares Luscious Lasagna with cashew cheese, salad with herb dressing, raw apple pie. Yum!

- Let's go ahead and add a block to this view. Go to `admin/build/views/browse_workshops/edit`. Check the **Provide Block** box and set the block **View Type** to **Date: Date Browser**.
- Add this block to a sidebar in `admin/build/block`.

The date browser module creates a small block with a single link for the current period (the period – Year, Month, Day is determined by the view argument option).

Upcoming Workshops

[Month of October 2009](#)

Recipe notes

- The **Date** module offers a number of theme functions for modifying output. Date 5.x-2.x helpfully has twice as many theme functions as Date 5.x-1.x. Regardless of which **Date** module version you are using, one good way to determine the function names and their associated filenames, is via the `grep` command. Log on to your host. Go to `sites/all/modules/date`, and enter the following command.

```
grep -r theme_ *
```

Alternatively, you may create a themeable functions reference in *Recipe 68*.

Recipe 40: Timeline

Ingredients

Timeline module: <http://www.drupal.org/project/timeline>
 Timeline widget: <http://www.miccolis.net/random/timeline-1-2.tar.gz>

The **Timeline** module implements a nice interactive script developed originally by the **SIMILE** project at MIT. The project "graduated" in 2008, to become its own stand-alone open source project, hosted at **Google Code**.

1. Log on to the host and enter the following commands. This is almost a garden-variety module installation, however, we also need to download all of the SIMILE timeline code and place it in an `api` subdirectory. The Timeline widget tarball expands to the **timeline-1-2** directory, so in the last step we use the `mv` command to rename it to `api`.

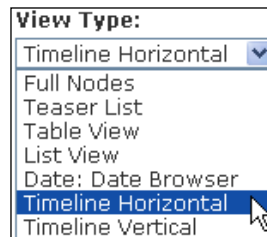
```
cd home/DRUPALACCOUNT/public_html/sites/all/modules
wget http://ftp.drupal.org/files/projects/timeline-5.x-1.0.tar.gz
tar xvf timeline-5.x-1.0.tar.gz
cd timeline/
wget http://www.miccolis.net/random/timeline-1-2.tar.gz
tar xvf timeline-1-2.tar.gz
mv timeline-1-2 api
```

2. Enable the **Timeline** module.

3. Explore the configuration settings at `admin/settings/timeline`. Here you may set the width, height, and initial date focus. Also note the **Advanced settings** for the path to the Timeline widget JavaScript. If, for some reason, you are unsuccessful making the widget available, you may temporarily try pointing to: `http://simile.mit.edu/timeline/api/timeline-api.js`. (This will work for as long as MIT continues to host this.)
4. Read the `README.txt` file, and `INSTALL.txt` file (this is always a good practice when installing new modules). This is a summary of the **Views** field requirements for setting up a timeline in Drupal.

	If present	If not present
First field	Date or Event determines location on the timeline	Defaults to Node Created Time
Second Field	Title, with clickable link	Node's default title is used
Third Field	Description	Teaser, if available.
Additional Fields	Additional fields are not used by Timeline.	

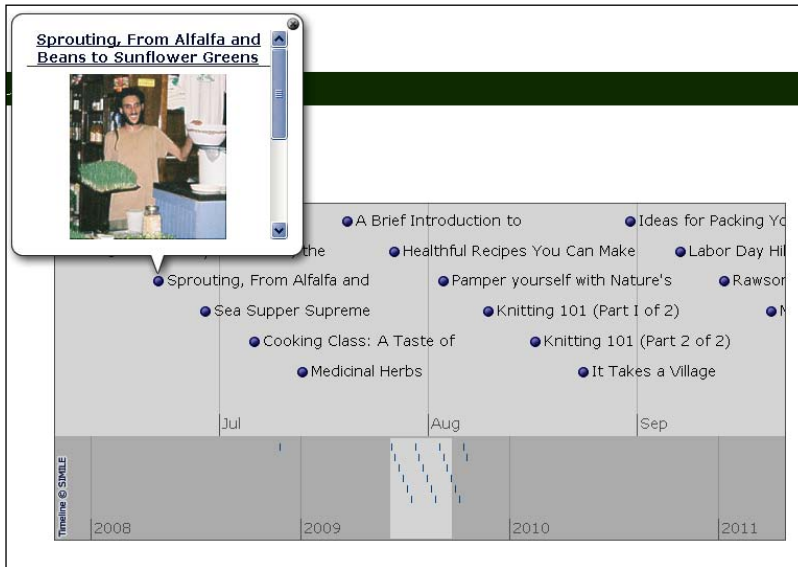
5. Edit the **timeline** view created in *Recipe 34*.
6. In the **Page** fieldset, change the **View Type** to **Timeline: Horizontal**.



7. Scroll down to the **Fields** fieldset. Note that in our original view, the **Title** is the first field followed by **Date** as the second field. We will reverse this. Click on the up arrow in the **Date** row to **Move this item to the top**.



8. Add the **Text: Description (field_workshop_description)** as the third field.
9. **Save** the view and browse to timeline.



Move the cursor around the timeline. Click on the events to see the associated pop-up box. Click and drag to the left to go forward in time. Click and drag works in both the upper and lower timeline bands.

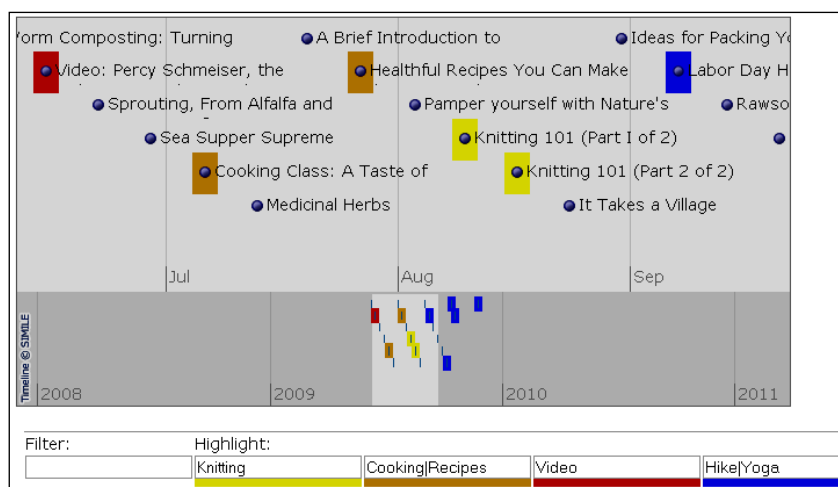
10. Configuration settings for individual timelines are available at `admin/build/timeline` (these settings will override default settings at `admin/settings/timeline`). A particularly interesting setting is the one to enable controls.

Enable controls:

☐ Disabled

☒ Enabled

With controls enabled, the user will have the opportunity to filter, or highlight the timeline based on text either in the title or the description.



Regular expressions will also work when filtering or highlighting timeline items.

Cooking Recipes	Find items that contain either Cooking or Recipes in the title or description
^L	Find titles that start with L
asagna	Find Lasagna – partial words work fine.
hea[rl]t	Find Health or Heart

The searches are case-insensitive.

Recipe notes

- For examples of basic and sophisticated SIMILE Timelines see <http://simile.mit.edu/timeline/examples/>.
- More information about Timeline is available at <http://code.google.com/p/simile-widgets/> including a mailing list, a wiki, and an independent issue queue for various SIMILE web widgets.
- There is also an archived listerv available at <http://www.nabble.com/SIMILE---General-f27660.html>. The listserv remains active for other SIMILE projects but does not accept content regarding "graduated" projects such as Timeline. However, you may find older posts helpful.

- If you like Timeline, try some of the other SIMILE projects as well, at <http://simile.mit.edu/>. (Perhaps you can write the next module to make them available in Drupal.)
- Exposed Filters are incompatible with the **Timeline** module.
- Arguments will work. For instance, if you add a Taxonomy item argument the URL: <http://YOURSITE.com/timeline/Cooking> would show only the workshops in the **Cooking** category.
- To incorporate color coding by taxonomy, consider installing the patch at <http://drupal.org/node/121298>. There is also a nice patch to include icons by taxonomy term at <http://drupal.org/node/104890>.
- The **Timeline** module integrates with the **Events** module.
- If no fields are included in the timeline, the timeline displays the date the content was posted.

Recipe 41: Views Popup

Ingredients:

Completed Recipe 34, `views_popup` view

Views Popup module: http://drupal.org/project/views_popup

The **Views Popup** module displays a list of up to three fields and then, upon mouseover, displays additional fields in a pop-up box. This is not a date-specific module, but we'll use it to display a workshop description when you move the mouse over the date. We'll also take a look at the `views_style_plugins` hook, which in this chapter enabled the **Date Browser**, **Timeline**, **Calendar**, and now **List View as Popup** View Types.

1. Install and enable the **Views Popup** module.
2. Open the **views_popup** view at `views_popup/edit`.
3. In the **Page** fieldset, change the **View Type** to **List View as Popup**, **Show 2 Fields**.
4. In the **Fields** fieldset, change the order of the **Datestamp** and **Title** fields so that the datestamp is first.

5. Add the **Text: Description (field_workshop_description)** field.

Name	Label	Handler	Option	Sortable	Default Sort
Datestamp: workshop_date (field_workshop_date)		Group multiple values	Default	No	None
Node: Title		Normal	As link	No	None
Display the title of the node.					
Text: Description (field_workshop_description)		Group multiple values	Default	No	None

The first two of these fields will display. The third will be hidden by default.

6. **Save** and view the **views_popup** view. Note than when you mouse over the date, the description appears.



7. Take a look at `hook_views_style_plugins` in the `views_popup.module` file:

```
function views_popup_views_style_plugins() {
  return array(
    'list_hint' => array(
      'name' => t('List View as Popup, Show 1 field'),
      'theme' => 'views_view_list_hint_popup1',
      'validate' => 'views_ui_plugin_validate_list',
      'needs_fields' => true,
      'weight' => -10,
    ),
    'list_hint2' => array(
      'name' => t('List View as Popup, Show 2 fields'),
    )
  );
}
```

```

        'theme' => 'views_view_list_hint_popup2',
        'validate' => 'views_ui_plugin_validate_list',
        'needs_fields' => true,
        'weight' => -10,
    ),
    'list_hint3' => array(
        'name' => t('List View as Popup, Show 3 fields'),
        'theme' => 'views_view_list_hint_popup3',
        'validate' => 'views_ui_plugin_validate_list',
        'needs_fields' => true,
        'weight' => -10,
    )
);
}

```

The critical elements in this array are the name and the specified theme callback function. The function appears below in the same file and does the actual work of producing the output, in this case, creating a pop-up.

Recipe notes

- Documentation on **hook_views_style_plugins** is available in the Drupal handbook at <http://drupal.org/node/193448>.
- Think of the style plugin as a way to modify the output of a whole node, while the field formatters that we saw in the previous chapter apply to a single field.
- *Appendix C* contains an extensive list of views style plugins (View Types) implemented by a more than sixty Drupal 5 modules.

Recipe 42: Creating a Calendar

Ingredients

Completed Recipe 34, Steps 1-9.

Calendar module: <http://www.drupal.org/project/calendar>

Calendar Popup module (Comes with calendar versions 5.x-2.x)

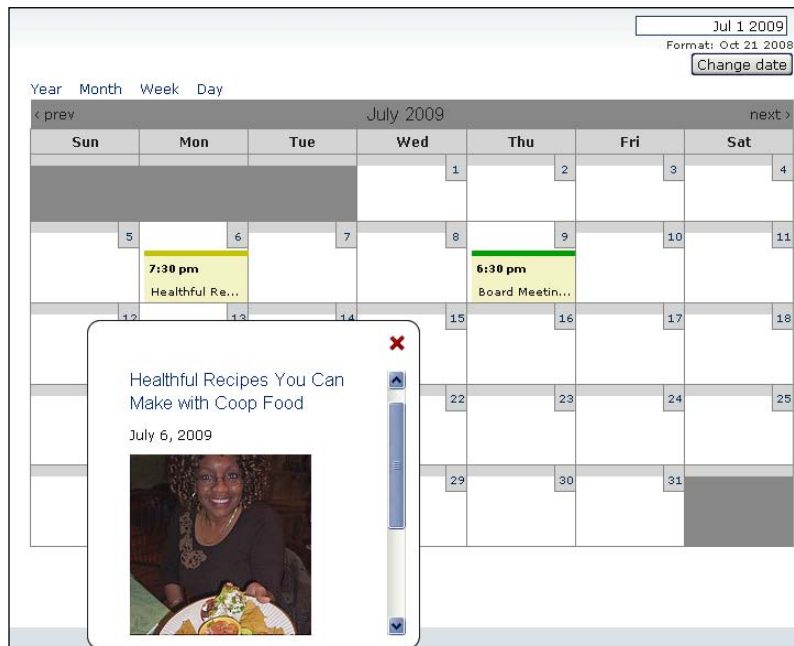
The **Calendar** module offers a wide variety of display options and makes three blocks available. The module comes with a helpful Default View, which we'll edit to display workshops.

Installation and configuration

1. Complete *Recipe 34*, Steps 1-9.
2. Install and enable the **Calendar** and **Calendar Popup** modules.

View setup

3. Visit `admin/build/views` and click on the **Add** link to add the calendar's default view. This pulls up the view edit form. Note that the **View Type** for both the **Page** and **Block** view is **Calendar**.
4. Note the three arguments to the view that appear in the following order: **Calendar Year**, **Calendar Month**, and **Calendar Day**. Each is set to **Display All Values** (if you create a calendar view from scratch, be sure to replicate the selection and order of these three fields).
5. Try saving the view. A warning appears noting that **The Calendar View requires at least one date field**. Go ahead and add the **Datestamp: workshop_date** field to the menu. A second warning appears noting that **Calendar CCK Date fields must be set to 'Do not group multiple values'**. Return to the **Fields** fieldset and set **Datestamp: workshop_date field** to **Do not group multiple values**. You may also add a second date or datestamp for another content type, at this time, if you have one.
6. Visit the calendar URL: `http://YOURSITE.com/calendar`.



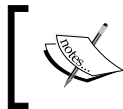
7. Configure the Calendar settings at `admin/settings/calendar` (note that these settings are only available after the first calendar view has been created). Choose your favorite time style, and turn off the display of week numbers.

Time format:
 
 The format to use for the time-only date display.

Show week numbers:
 

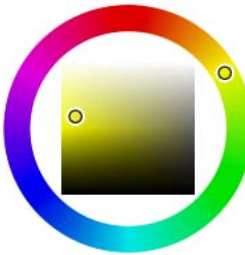
8. Explore the other settings on this page as well. Ordinarily, they will be left as the default setting. Click the **Submit** button.
9. Click on the **Legend** tab. Note that the module can color calendar events by content type. Setting a color is a three-click process:
- Click in the edit box for the workshop content type after the `#ffffff` (that's the hex code for the color white).
 - Click in the color circle. This will create a palette that blends the selected color with a range of black and white.
 - Click in the color **Swatch**.

If you have more content types that you wish to add to the calendar (they must include a date field or an event) repeat these steps for each one.



If you select the whole `#ffffff` text rather than single-clicking in the edit box, the color swatch may be applied to the wrong content type depending on your browser.

Legend colors:

	Board Meeting: #3afd3b	Chord: #ffffff
	Image: #ffffff	Meeting: #ffffff
	Page: #ffffff	Story: #ffffff
	Timeline Item: #ffffff	Workshop: #f5f538

Installing Calendar Blocks

10. Go to `views/build/blocks` and place all three calendar blocks in the sidebar.

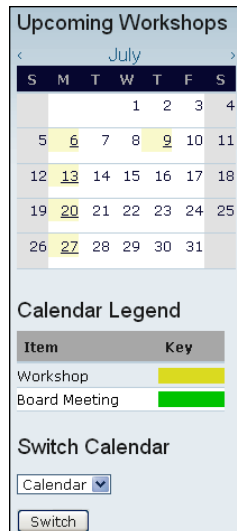
Calendar Blocks

calendar

Calendar Legend

Calendar Switcher

11. Go to `http://YOURSITE.com/calendar` and look at the sidebar.



The greater-than (>) sign in the calendar, of course, will switch the mini-calendar to the next month.

Notice the URL parameters for `http://YOURSITE.com/calendar/2009/07?mini=calendar/2009/07/all`.

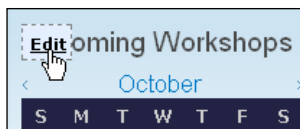
(The parameters may offer a quicker way of navigating the mini-calendar.)

The calendar legend collapses and expands based on the number of content types shown on the page calendar.

The **List**, **Teasers**, **Nodes**, and **Table** switch options offer navigation similar to the Date Browser View type from *Recipe 39*.

Of the three blocks, two (the Legend and the Switcher) are designed to appear only on the calendar page (that is, when the URL is `calendar`). We will now set the mini-calendar block to *NOT* appear when the calendar is displayed.

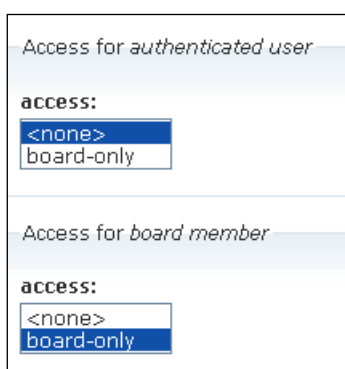
12. If you happen to be using a theme based on Zen, Basic, or Paintbrush, helpful edit block links appear when you mouse over any block.



13. Click on the **Edit** link. Alternatively, if mouseover Edit links are not available, visit `admin/build/block` and search for the correct block.
14. In the **Page specific visibility settings** fieldset, make sure that the **Show on every page except the listed pages** option is selected. Set the **Pages** to **calendar***. The asterisk ensures that the mini-calendar will be excluded on any page that has **calendar** in the URL, including URLs with arguments and parameters.

Recipe notes

- Check the handbook page on the calendar module, at <http://drupal.org/handbook/modules/calendar>.
- A common practice is to create a custom sidebar block containing an attractive calendar graphic with a link to `YOURSITE.com/calendar`.
- If you want some calendar items to display only for designated roles, try installing an access control module such as **Taxonomy Access Control Lite**, (http://drupal.org/project/tac_lite). All board meetings, for instance, could be given a taxonomy term of **board-only** and then the **board member** role could be granted access to that term. Views, including the calendar, happily honors these access controls.



- The calendar module comes with an extensive `calendar.css` file which can be used to customize the calendar colors and other attributes. I recommend several resources:
 - Firebug, of course, for identifying the class names, available at <http://getfirebug.com/>
 - ColorZilla, another helpful firebug extension, available at <https://addons.mozilla.org/en-US/firefox/addon/271>. Click on an object on screen and the color is revealed. You may then copy the hex code and search for it in the CSS file.
 - Another good way to identify all the configurable calendar colors is to search `calendar.css` for the `#` character.

```
/* The inner 'node' displayed in the full month view */
.calendar-calendar td .inner div,
.calendar-calendar td .inner div a {
background-color:#fff;
color:#000;
}

.calendar-calendar div.date-nav {
background-color:#ccc;
color:#777;
padding:0.2em;
width:auto;
border:1px solid #ccc;
}
```

- Copy the rule changes to your theme's `style.css` file.
- If you edit the module's `calendar.css` file directly, consider installing the **Site notes** module <http://drupal.org/project/sitenotes>. Note your CSS changes so that next time you upgrade the calendar module, you will have a record.

Recipe 43: iCal feeds

[Ingredients]
Completed Recipes 34 and 42, calendar view
iCal module, which comes with Calendar

iCal is a standard text format for exchanging calendar data between web sites. The Drupal Calendar module enables two-way exchange: it can import and export calendar items, that is, calendar events appearing on an external site can be displayed on the Drupal site, and events from the Drupal calendar page may appear on an external site.

Import Calendar items

1. Enable the iCal module that comes packaged with the Calendar module.
2. Identify the iCal feed from a web site, and copy the iCal URL. In most browsers you may right-click, and select **Copy link location**. Of course, this is only possible if the web site provides an iCal feed.



3. Configure the iCal settings at `admin/settings/calendar/calendar/ical`. The path to get there is **Administer | Site Configuration | Calendar Setup | iCal**. Enter a name for the feed, the iCal url, and a Default link.

▼ Feed 0

Name:

Baltimore Women's Film Festival

The name of a feed to include in this calendar.

iCal url:

http://baltimorewomens.bside.com/calendar/festival/baltimorewomens2008.ics

The ical url, either an external feed url or internal file path and name. Change 'webcal:/' to 'http:/'.

Default link:

http://baltimorewomens.bside.com/2008/schedule/2008-10-26

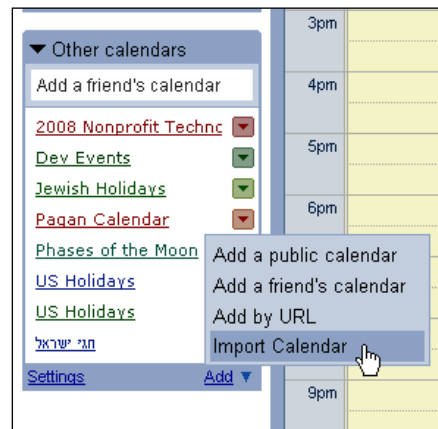
A url to use to link back to ical items without a url, generally the public link to the calendar these items are coming from.

- Open the Calendar view and note the new events from the feed.

19	20	21	22	23	24	25
			6:00 pm - 7:30 pm Short Dramatic Films: Parents and Children	11:30 am - 12:00 pm Patricia Baltimore 12:30 pm - 1:00 pm Til Death do us part 2:00 pm - 3:00 pm Beyond Breast Cancer: Stories Of Survivors	11:00 am - 11:30 am Two Angry Moms 1:00 pm - 2:24 pm Who Does She Think She Is? 2:45 pm - 4:15 pm Animation and Music Videos	

Export Calendar items

- Edit the calendar view at **calendar/edit**. Add a fourth argument of **Calendar: iCal Feed** and set the **Default** to **Display All Values**.
- Go back to the calendar view and notice the little calendar icon .
- Click on the icon and download the iCal file.
- Go to the calendar service of your choice and upload this link. We will use Google Calendar in this example. Go to Google Calendar at calendar.google.com and click the **Add** link that appears below the **Other calendars**. Choose **Import Calendar**.



9. Select the iCal file (it has a `.ics` extension) and the calendar where the events should be saved. Your Drupal Calendar events will now appear on the Google Calendar.

Add Other Calendar

[Friends' Calendars](#) [Add by URL](#) **Import Calendar**

Step 1: Select File [Learn more](#)

 Choose the file that contains your events. Google Calendar can import event information in iCal or CSV (MS Outlook) format.

Step 2: Choose Calendar

 Choose the calendar where these events should be saved.

Step 3: Complete Import

[« Back to Calendar](#)

Recipe notes

- Wikipedia has a helpful entry describing the iCal format
<http://en.wikipedia.org/wiki/ICalendar>.

[ Distinguish that URL from <http://en.wikipedia.org/wiki/ICal>, which is not about the iCal format, but rather about an Apple software product that uses the format.]

- A number of issues have been posted to the issue queue regarding iCal support in Drupal 5.x and 6.x. You may follow or participate in the issue queue at <http://drupal.org/project/issues/calendar> (see *Recipe 88* for information on how to search the issue queue).
- Drupal developer Ceardach offers a helpful alternative to using the calendar iCal module (<http://ceardach.com/blog/2008/06/customized-ical-export-drupal>).
- Imported Drupal 5 iCal feeds point to an off-site link specified in the configuration. In Drupal 6, by contrast, date data will actually be imported into the Drupal site as separate nodes, so calendar item links will point to them.

Recipe 44: Upgrade Date and Calendar modules

Ingredients

Date 2.x module: <http://drupal.org/project/date>

Calendar 2.x module: <http://drupal.org/project/calendar>

Backup and Migrate module (Optional): http://drupal.org/project/backup_migrate

There are some pronounced differences between 1.x and 2.x versions for both the **Date** and **Calendar** modules. The details are documented in the release notes at <http://drupal.org/node/70963/release>, and <http://drupal.org/node/92594/release>.

This recipe is applicable if you are currently using **Calendar 5.x-1.x** and **Date 5.x-1.x** and wish to upgrade to **Calendar 5.x-2.x** and **Date 5.x-2.x**. It is also more widely applicable to other module upgrades.

The general practice for upgrading Drupal and its contributed modules is to replace the old code with the new and visit <http://YOURSITE.com/update.php>. However, there are prerequisite steps to ensure success. Back up the web site first (including both the files and the database). Be sure that the backups are not in a web-accessible location. For instance, if your web server is located in the `public_html` directory, ensure that none of your backup files, especially the SQL backup are saved there or in one of its subdirectories.

The process for performing upgrades is described in `UPGRADE.txt` in the Drupal site directory. The guidance below follows the same steps, but provides more detail.

Backing up the Database

There are several database backup approaches. We'll now look at two approaches (capital letters signal variables that differ according to the site).

Approach 1: Backup with mysqldump

```
mkdir /backups
```

```
mkdir /backups/MYSITENAME
```

```
cd /backups/MYSITENAME
```

```
mysqldump --opt -u MYUSERNAME -p MYDBNAME > mysitenam_backup.sql
```



If you have forgotten your database name, password, and/or username, check the `sites/default/settings.php` file. Look for the `$db_url` variable in the **Database settings** section. It will have this format:

```
$db_url = 'mysql:// MYUSERNAME:PASSWORD@localhost/ MYDBNAME'
```

If you are not confident that your backup directory is secure on your server, then FTP the files to your local hard drive, and remove them from the server.

Approach 2: Backup with the Drupal Backup and Migrate module

Install the **Backup** and **Migrate** module.

The configuration options are found in the **Administer | Content Management | Backup and Migrate** (you may be surprised at first not to find the link in the Site Configuration area). If you choose to save the SQL file to the files directory, be sure to move it from there as soon as you are done. The module makes an effort to edit the `.htaccess` file of your website, to try to ensure protection, but the module's own `README.txt` file notes that "this is not guaranteed to work on all environments". Read the file for more security guidance.

Exclude the following tables altogether:

access
accesslog
aggregator_category
aggregator_category_feed
aggregator_category_item
aggregator_feed
aggregator_item
authmap
blocks
blocks_roles

The selected tables will not be added to the backup file.

Exclude the data from the following tables:

access
aggregator_category
aggregator_category_feed
aggregator_category_item
aggregator_feed
aggregator_item
authmap
blocks
blocks_roles

The selected tables will have their structure backed up but not their contents. This is useful for excluding cache data to reduce file size.

Backup file name:

Compression:

☒ None
☐ GZip
☐ BZip

Destination:

☒ Download
☐ Save to Files Directory

☒ Append a timestamp.

Timestamp format:

Should be a PHP `date()` format string.

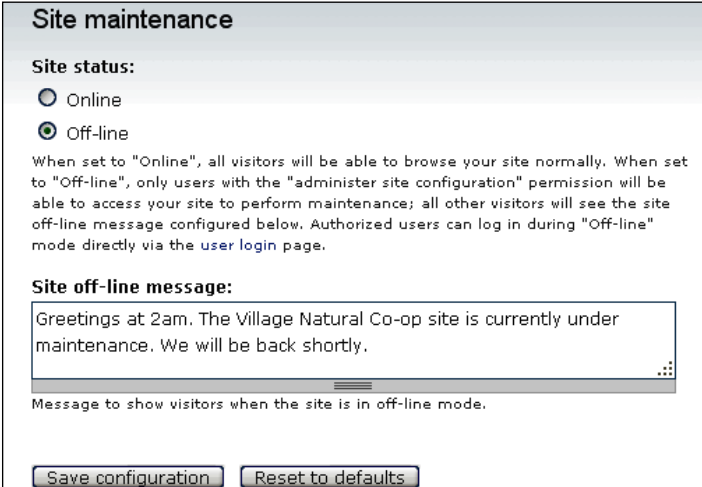
☒ Save these settings.

Upgrading Calendar and Date

1. Back up the database using one of the two approaches above (it never hurts to reiterate the value of backups – and they are easy to perform).
2. Backup the **sites** folder and the **files** folder, as well as the **.htaccess** and **robots.txt** files, as follows.

```
mkdir /backups
mkdir /backups/MYSITENAME
cd /backups/MYSITENAME
cp -r /home/<DRUPALROOT>/public_html/sites .
cp -r /home/<DRUPALROOT>/public_html/files .
cp -r /home/<DRUPALROOT>/public_html/.htaccess .
cp -r /home/<DRUPALROOT>/public_html/robots.txt .
```

3. Log on to the website as the super-administrator with the user ID 1.
4. If this is a live site (rather than just a development server), place the site off-line, at `admin/settings/site-maintenance`.



The screenshot shows the 'Site maintenance' configuration page. At the top, the title 'Site maintenance' is displayed. Below it, the 'Site status:' section has two radio buttons: 'Online' and 'Off-line'. The 'Off-line' button is selected. A paragraph of text explains the difference between 'Online' and 'Off-line' modes. Below this, the 'Site off-line message:' section contains a text area with the message: 'Greetings at 2am. The Village Natural Co-op site is currently under maintenance. We will be back shortly.' At the bottom of the form are two buttons: 'Save configuration' and 'Reset to defaults'.

5. Disable all the **Date** and **Calendar** modules at `admin/build/modules` (including any modules that are dependent on **Date** or **Date API**, or **Calendar**). Check for modules in the following fieldsets, at a minimum: **CCK**, **Other**, and **Views**. You may need to press the **Save configuration** button several times, to disable modules that are dependent on other modules.
6. Any time you perform a Drupal update it's a good idea to switch to the Garland theme first. Change the general theme at `admin/build/themes` and the administration theme at `admin/settings/admin`.

7. Delete the calendar and date module subdirectories, including all of its contents.
8. Download and unpack the new calendar and date modules into `sites/all/modules`.

Implement Steps 7-8 via the following commands (determine the current module links for the **wget** commands by visiting <http://drupal.org/project/date> and <http://drupal.org/project/calendar>).

```
cd /home/<DRUPALROOT>/public_html/sites/all/modules
rm -rf date
rm -rf calendar
wget http://ftp.drupal.org/files/projects/date-5.x-2.3.tar.gz
tar xvf date-5.x-2.3.tar.gz
wget http://ftp.drupal.org/files/projects/calendar-5.x-2.3.tar.gz
tar xvf calendar-5.x-2.3.tar.gz
```

9. Run `update.php` by visiting <http://YOURSITE.com/update.php>.
10. The following screen appears. Click on the **run the database upgrade script** link.

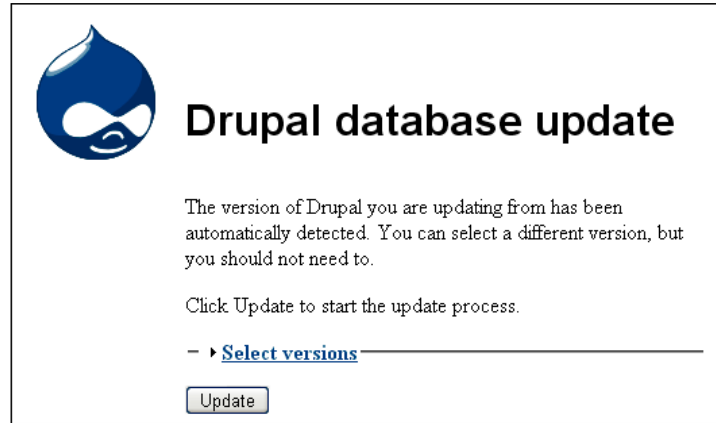


Drupal database update

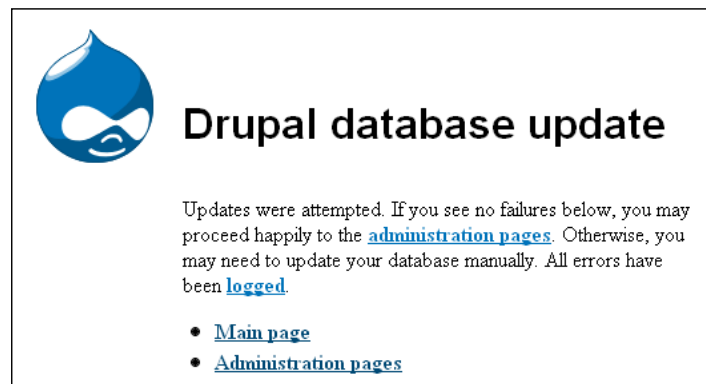
1. Use this script to **upgrade an existing Drupal installation**. You don't need this script when installing Drupal from scratch.
2. Before doing anything, backup your database. This process will change your database and its values, and some things might get lost.
3. Update your Drupal sources, check the notes below and [run the database upgrade script](#). Don't upgrade your database twice as it may cause problems.
4. Go through the various administration pages to change the existing and new settings to your liking.

For more help, see the [Installation and upgrading handbook](#). If you are unsure what these terms mean you should probably contact your hosting provider.

11. In the next screen, simply click on the **Update** button.



A quick progress bar appears followed by the screen below. At first words like "attempted" and "failures" may look scary. However, upon careful reading, this screen simply offers links to the site and lets you know that "you may proceed happily". Should you run into error messages at this point (which almost never happens) you will need to restore your backup.



12. Click on the link to the **Administration pages**, and go to `admin/logs/status`. Also make a note of the PHP version you use.

13. Go to `admin/build/modules`. Note that there is a new **Date/Time** fieldset containing the updated modules. Enable **Calendar**, **Calendar Popup**, **Date**, **Date API**, **Date Popup**, and other modules as desired. If you are using PHP version 5.1 or below, be sure to enable the **Date PHP4** module.

Date/Time			
Enabled	Name	Version	Description
☐	Calendar	5.x-2.3	Views plugin to display views containing dates as Calendars. Depends on: Views (enabled), Date API (disabled), Date Timezone (disabled) Required by: Calendar iCal (disabled), Calendar Popup (disabled)
☐	Calendar iCal	5.x-2.3	Adds ical functionality to Calendar views. Depends on: Views (enabled), Date API (disabled), Calendar (disabled)
☐	Calendar Popup	5.x-2.3	Replaces the links to calendar items with a javascript popup that gracefully regresses if javascript is not enabled Depends on: Calendar (disabled), Views (enabled)
☐	Date	5.x-2.3	Defines CCK date/time fields and widgets. Depends on: Content (enabled), Date API (disabled), Date Timezone (disabled) Required by: Date Copy (disabled)
☐	Date API	5.x-2.3	A Date API that can be used by other modules. Required by: Calendar (disabled), Calendar iCal (disabled), Date (disabled), Date PHP4 (disabled), Date Popup (disabled), Date Repeat API (disabled), Date Timezone (disabled)
☐	Date Copy	5.x-2.3	Import and export CCK date data. Depends on: Content (enabled), Date (disabled)
☐	Date PHP4	5.x-2.3	Emulate PHP 5.2 date functions in PHP 4.x, PHP 5.0, and PHP 5.1. Required when using the Date API with PHP versions less than PHP 5.2. Depends on: Date API (disabled)
☐	Date Popup	5.x-2.3	Enables jquery popup calendars and time entry widgets for selecting dates and times. Depends on: Date API (disabled), Date Timezone (disabled)
☐	Date Repeat API	5.x-2.3	A Date Repeat API to calculate repeating dates and times from iCal rules. Depends on: Date API (disabled)
☐	Date Timezone	5.x-2.3	Needed when using Date API. Overrides site and user timezone handling to set timezone names instead of offsets. Depends on: Date API (disabled) Required by: Calendar (disabled), Date (disabled), Date Popup (disabled)

14. Visit `admin/build/themes` and `admin/settings/admin` and set the public and administration themes back to their original settings.
15. If this is a live site, visit `admin/settings/site-maintenance` and bring the site back online.

Recipe notes

- Be sure to read the `UPGRADE.txt` file in the Drupal installation. Additionally, the following pages offer helpful guidance.
 - <http://drupal.org/upgrade/running-update-php>
 - <http://drupal.org/upgrade>
- Consider installing the **Update Status** module from, http://drupal.org/project/update_status. Like the **Views** module, **Update Status** was written by Earl Miles (merlinofchaos). The module provides alerts about both Drupal core and module updates. Updates are important: they often fix security vulnerabilities. See *Recipe 53* for information on how to set up cron, which is required for the **Update Status** module to work.

Summary

This chapter focused on just a few modules, with an emphasis on date and calendar. We explored date formats noting a broad range of Drupal locations where the special formatting for PHP dates can be applied. We created a view of **Upcoming Workshops** that served as the foundation for the next nine recipes. We explored (finally!) the **Exposed filters** fieldset including some special filters enabled by the Views **Date Range Filter** Module. We had a chance to create some summary views. We made use of four special **Views Types** coded in modules via the `hook_views_style_plugins`. These interesting View Types include the **Date Browser**, **Timeline**, **List View as Popup**, and **Calendar** (*Appendix C* contains a comprehensive list of Views style plugins and their associated View Types). Finally, we looked in detail at the steps for upgrading from the older 5.x-1.x **Calendar** and **Date** modules to the newer, and more feature-rich, 5.x-2.x versions.

5

Views and Tools for Administrators

In *Recipe 7* in Chapter 1, we installed the **Admin Role** module ensuring that the role named **admin** has access to all the modules' permissions. In this chapter, we will see a wide variety of additional modules and views to ease the work of the web site administrator.

Recipe 45: Administration menu

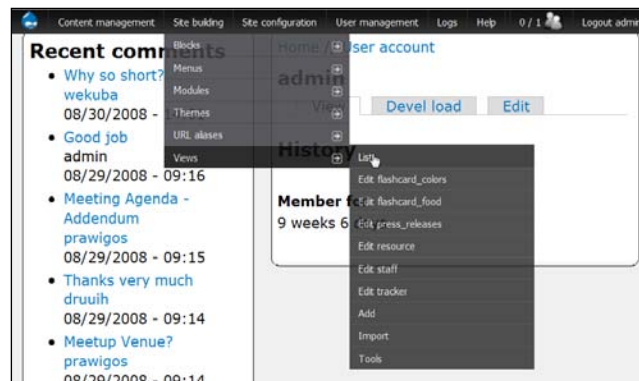
Ingredients

Administration menu module:

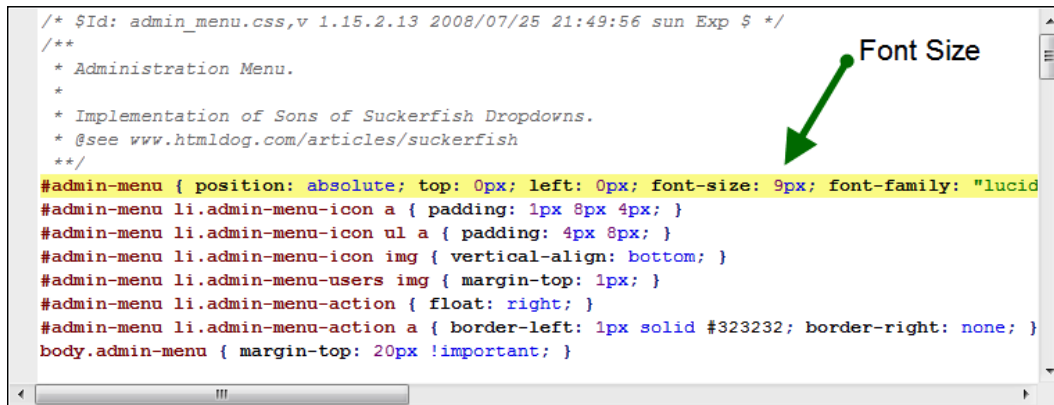
http://drupal.org/project/admin_menu

The **Administration menu** is a fantastic module, easing the administration of any Drupal web site. It's an install-and-go module, but there are a few features worth knowing about.

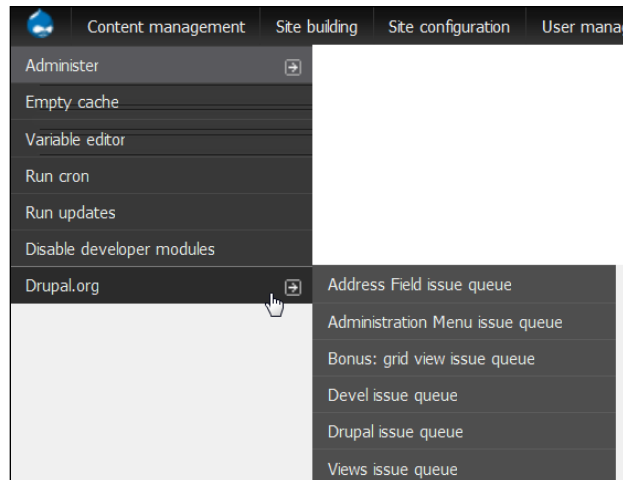
1. Install and enable the **Administration menu** module.



2. A new drop-down menu appears at the top of the page. Mouseover and click on the various options to become familiar with it.
3. If you find the menu font size to be a bit small for comfortable reading, open the `admin_menu.css` file in the `sites/all/modules` directory. Edit the **font-size** in the first line.



4. The **Administration menu** has more detailed help than most contributed modules. Access it at `admin/help/admin_menu`.
5. Note the special options available upon clicking the Drupalicon (the Drupal logo). If you have the **display drupal links** permission enabled for your role, links appear for each enabled module to the `drupal.org` issue queue.



6. You should also note that if the **Devel** module is enabled, there is a handy link to clear the cache (a real time-saver) along with some other related links.

Recipe notes

- The **Administration menu Dropdown** module http://drupal.org/project/admin_menu_dropdown is a nice companion to the Administration menu module. Instead of displaying at the top of the page all the time, the menu appears at the cursor location upon pressing *Ctrl + Alt*.
- The `admin_menu.inc` file contains a helpful theme function which you may override to change the icon. If you want a quick way to create the `favicon.ico` file, consider using the **Favicon Generator** service: <http://antifavicon.com/>.

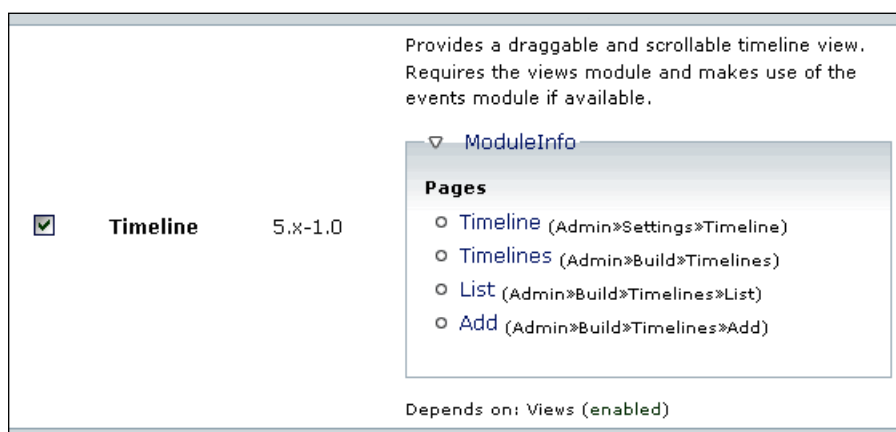
Recipe 46: ModuleInfo

Ingredients

ModuleInfo module: <http://drupal.org/project/moduleinfo>

Drupal users often play a frustrating version of "hide and seek" when installing a new module. "Okay, it's installed, now where do I go?" The **ModuleInfo** module answers that question on the module administration page.

- Install and enable the **ModuleInfo** module.
- Go to the `admin/build/modules` page and note the new **ModuleInfo** fieldsets. Voilà! Now there are links to each related page, block, and content type:



Recipe notes

- This module may remind you of long forgotten links. Do you remember the **Views UI Help** page that we looked at in *Recipe 2*? Now is a good time to look at it again (roles with the **access administration pages** permission will be able to see Help).

Recipe 47: Views UI permissions

Ingredients

Views UI permissions module:
http://drupal.org/project/views_ui_perm

Masquerade module: <http://drupal.org/project/masquerade>

There is no doubt about it: The Views UI is powerful. Given its complexity, many site administrators will benefit from a pared down interface. The **Views UI Permissions** module restricts access by role to selected fields and fieldsets in the views interface. The **Masquerade** module will facilitate our work, as we switch (temporarily) from one user to another, while testing this functionality.

1. Install and enable the **Views UI Permissions** and **Masquerade** modules.
2. Go to `admin/settings/views_ui_perm`. For this example, leave the default settings intact and add the following:

Section	Enable
Page Permissions	Page nodes per page
Block Permissions	Block nodes per block
Other Permissions	Fields
	Sort Criteria

3. Note at the top of this page the direction to read the module's `README.txt` file (it is always a good idea to read the `README.txt` file). This module's file, located at `sites/all/modules/views_ui_perm/README.txt`, provides guidance for editing the theme to enable **Views UI Permissions**. Go ahead and follow the instructions in the file. Be sure not to miss the last curly bracket when copying the `phptemplate_fieldset` into `template.php`. `template.php` is found in the `sites/all/themes/<YOURTHEME>` directory. If you use separate public and administrative themes be sure to copy the function to both `template.php` files.
4. Go to `admin/user/roles` and add a role named **editor**.

5. Go to `admin/user/user/create`. Add a user named **editor** and assign this new user the role of an editor.

Roles:

☐ admin

☒ editor

6. Set the permissions for the **editor** role at `admin/user/access`. Provide these permissions at a minimum. Do not enable the **views_ui – administer views** UI permission for the editor role.

Module	Permission to enable
admin_menu (If installed)	access administration menu
block	administer blocks
node	access content
views	access all views
views_ui_perm	restricted views ui

7. Configure the **Masquerade** module at `admin/settings/masquerade`. Set the role that is considered administrator to **admin**. The configuration adds a single switch user option to the Administration menu, and also allows selection of additional users in a block.

Roles that are considered 'administrator' for masquerading.:

☒ admin

☐ anonymous user

☐ authenticated user

☐ editor

Default test user:

editor:

Enter the username of an account you wish to switch easily between via a menu item. The name must be an existing user.

Quick switch users:

admin

editor

user1

Select the usernames for accounts to show as quick links in the Masquerade block.

masquerade.module version: 1.15.2.12 , 2008/09/25 12:09:34

- Place the **Masquerade** block in a region of the page (typically one of the sidebars) at `admin/build/blocks`.



The screenshot shows a 'Masquerade' block with a text input field for a username, an 'Enter' button, and a 'Go' button. Below the input field, it says 'username to masquerade.' and 'Quick switches:' with radio buttons for 'editor' and 'user1'.

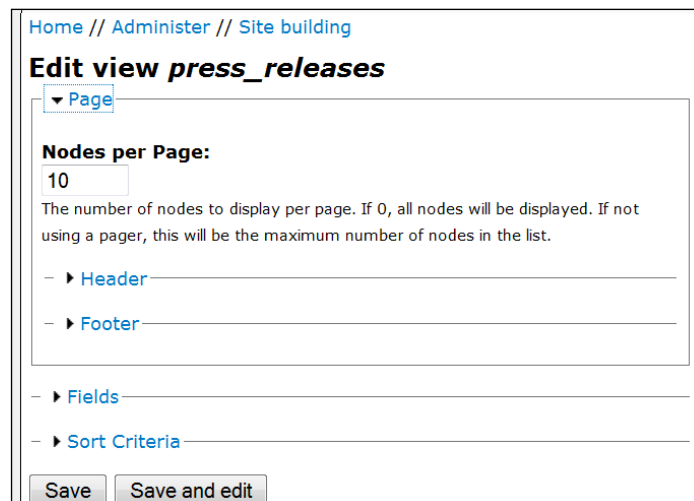
- Now, to prepare for the moment of truth, switch to the user named **editor** (or any user who has the editor role). Note that the **Masquerade** block has changed to a **Switch back** link. We will use this later.

[ Our editor role has very restricted access. Depending on the page you are on when you switch to editor, you may see an Access Denied message. However, you can still navigate to a view with our editor permissions.]



The screenshot shows the 'Masquerade' block after switching to the 'editor' role. It displays 'You are masquerading as: editor' and a 'Switch back' link.

- Edit a view, for instance, **press_releases** at `http://YOURSITE.com/press-releases/edit`.



The screenshot shows the 'Edit view press_releases' page. It includes a breadcrumb trail 'Home // Administer // Site building', a 'Page' dropdown, and a 'Nodes per Page' section with a value of 10. Below this are expandable sections for 'Header', 'Footer', 'Fields', and 'Sort Criteria'. At the bottom are 'Save' and 'Save and edit' buttons.

Note that the Views options that we did not enable are excluded. The **Block** fieldset also does not appear on this screen because no block was originally enabled for that view.

Save the view and visit the editor's View administration screen at `admin/build/views_ui_perm`. It is a pared-down version without links to the default views or import and tools buttons.

11. Having confirmed the successful modification of the editor's permissions, click on the **Switch back** link in the **Masquerade** block.

Recipe notes

- This **Views UI Permissions** module does not offer the ability to customize separate permissions for different roles.
- Here are some helpful URLs to memorize, for cases when the login box and Drupal **Administration Menu** are not available on a site.

	Clean URLs enabled	Clean URLs not enabled
Login	<code>example.com/user</code>	<code>example.com/?=user</code>
Logout	<code>example.com/logout</code>	<code>example.com/?=logout</code>

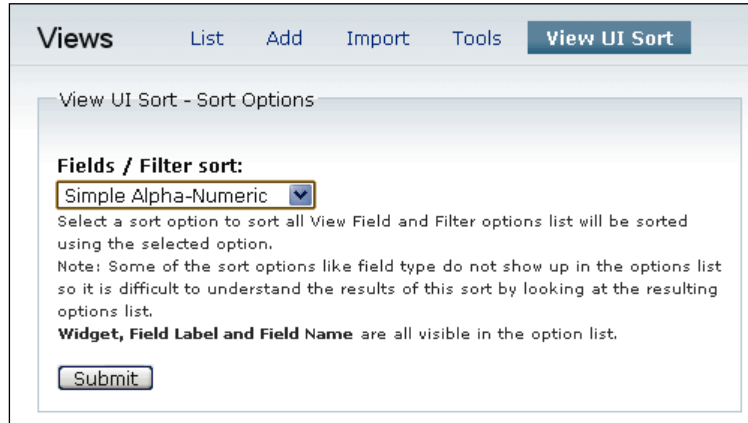
Recipe 48: Views UI Sort

[Ingredients]
Views UI Sort module: http://drupal.org/project/views_ui_sort

The **Views UI Sort** module addresses a long-time annoyance for Views administrators: that field names are not alphabetized in the user interface.

1. Install and enable the **Views UI Sort** module.
2. Go to `admin/build/views`, and notice the new tab available: **Views UI Sort**. Click on it, and set the **Fields/Filter sort** to **Simple Alpha-Numeric**.

(This configuration item is also available at `admin/settings/view_ui_sort`.)



Views List Add Import Tools **View UI Sort**

View UI Sort - Sort Options

Fields / Filter sort:

Simple Alpha-Numeric

Select a sort option to sort all View Field and Filter options list will be sorted using the selected option.

Note: Some of the sort options like field type do not show up in the options list so it is difficult to understand the results of this sort by looking at the resulting options list.

Widget, Field Label and Field Name are all visible in the option list.

Submit

3. Pull up a view, and enjoy the newly alphabetized **Add Field** selection lists in the **Field** and **Filter** fieldsets.

Before UI Sort	After UI Sort
Add Field Comment: Subject Node: Total Hits Node: Recent Hits Node: Last Hit Time Taxonomy: All Terms Taxonomy: Terms for Level Taxonomy: Terms for topic Taxonomy: Term Taxonomy: Term Description File: Id File: All files File: Name File: Path File: Size File: Mime type Node: Author Name User: Author Picture Embedded Video: Video: (field_url) Link: websnpr (field_websnpr) Node: Status Views Custom field: Text	Add Field Comment: Add link Node: Created Time Node: Delete link Node: Edit link Node: ID Node: Last Hit Time Node: Link to node Node: Recent Hits Node: Status Node: Title Node: Total Hits Node: Type Node: Updated Time Node: View link Taxonomy: All Terms Taxonomy: Term Taxonomy: Term Description Taxonomy: Terms for Level Taxonomy: Terms for topic User: Author Picture Views Custom field: Text

Recipe notes

- Experiment with other sort options to determine if you find a preference. Note that only the widgets, field labels, and field names are visible in the drop-down list so some of the sorts based on field types may not feel too intuitive.

Recipe 49: Editablefields

Ingredients

editablefields module: <http://drupal.org/project/editablefields>

The **editablefields** module is a tremendous time-saver for site administrators, making selected fields easily editable in a view. The module adds new view types and a new field option to the Views UI.

1. Install and enable the **editablefields** module.
2. Clone a view you wish to edit. Select the view to clone at `admin/build/modules`.

Fieldset	Action
Basic Information	Modify the view name. Example: admin_swim_groups Ensure that the view may only be seen by the admin role (see <i>Recipe 7</i> for guidance on the admin role, and associated Admin Role module).
Page	Ensure that Provide Page View is selected. Change the URL. Example: swim-groups-instructor-edit Change the View Type to Editablefields – table – No Form
Fields	Add the Node:Title (if not already there) and any additional fields you wish to see in the view. Set the field Option to Editable as desired.

3. **Save** the view. The view enables rapid updating of fields across multiple nodes.

☐ Water Wrigglers (Ages 2 to 3)

Instructor:

☐ Tadpoles (Ages 3 to 5)

Instructor:

☐ Guppies (Ages 6 to 12)

Instructor:

☐ Flying Fish (Ages 9 - 14)

Instructor:

☐ Adult Lessons (Age 15+)

Instructor:

Recipe notes

- An alternative to the **editablefields** module is the **Editview** module at <http://drupal.org/project/editview>.

Recipe 50: Views Bulk Operations— Content administration

Ingredients

Completed Recipe 7, Admin Role

Views Bulk Operations module:
http://drupal.org/project/views_bulk_operations

Views Bulk Operations (VBO) could just as well be called the "Better Site Administration" module. Let's start by looking at the content management page that comes with Drupal core at `admin/content/node`.

Content

Show only items where

☒ status is

☐ type

☐ category

Update options

☐ taxonomy_term

☐ Drupal issue bookmark

☐ Admin Links

☐ Function Reference for your site

☐ jQuery bookmarklet to expand all fieldsets

Type	Author	Status	Operations
Blog entry	drupal tips	published	edit
Blog entry	drupal tips	published	edit
Blog entry	drupal tips	published	edit
Blog entry	drupal tips	published	edit
Blog entry	drupal tips	published	edit

The form's user experience is a bit clunky, often requiring multiple clicks of the **Filter** button. In this recipe we provide an alternative to this form (in *Recipe 51* we'll incorporate new actions into the options list).

1. Install and enable **Views Bulk Operations**.
2. Go to `admin/build/views` and notice the new default view (core default views were covered in Chapter 2).

Default Views						
Default view	Title	Description	Provides	URL	Status	Actions
admin_content	Content	View, edit and delete your site's content.	Page	admin/content/node2	Enabled	Add Disable

3. Click on the **Add** link. Note that if you save this view without making any changes, you get a warning that the **Node: Type must have a value!**

Node: Type must have a value!

4. Before selecting the **Node: Type** filters, we will make an essential change to the default **Views Bulk Operations** view. Make sure that Views Bulk Operations are available only to the **admin** role (or other roles you wish to designate with key administrative privileges).

Access:

- ☒ admin
- ☐ anonymous user
- ☐ authenticated user
- ☐ editor

5. Go to the **Filters** fieldset and select all available values for the **Node: Type**.

The screenshot shows the 'Filters' fieldset in a Drupal interface. It contains a table with columns: Field, Operator, Value, Option, and Ops. The 'Field' column has 'Node: Type'. The 'Operator' column has 'Is One Of'. The 'Value' column has a dropdown menu open, showing a list of node types: 'Blog entry', 'Chat room', 'Page', 'Story', and 'Video'. The 'Option' and 'Ops' columns are empty. Below the table, there is a text label: 'Include or exclude nodes of the selected types.'

6. **Save** the view.
7. Check out your new content administration view at `admin/content/node2`. Upon loading the first time, the **Operations** form appears offering checkboxes of available actions.

Bulk operations for view *admin_content*

View Edit Clone Export Dev load Dev queries Dev items **Operations**

No operations are enabled. You must enable one or more operations in order to use Views Bulk Operations.

These settings determine which actions will be available from within a bulk operations view form. Operations selected here will be available for execution on the form. If only one action is selected, no dropdown will be provided and users will be able to perform or configure the single operation directly from the view form.

Note: Install the [Actions](#) module to enable additional operations.

- ☒ Delete (*views_bulk_operations_delete_nodes*)
- ☒ Demote from front page (*node_operations_demote*)
- ☒ Make sticky (*node_operations_sticky*)
- ☒ Promote to front page (*node_operations_promote*)
- ☒ Publish (*node_operations_publish*)
- ☒ Reload Embedded Media Data (*emfield_operations_reload*)
- ☒ Remove stickiness (*node_operations_unsticky*)
- ☒ Unpublish (*node_operations_unpublish*)

8. Go ahead and select all the operations and save the configuration.

Content

View Edit Clone Export Dev load Dev queries Dev items Operations

Type	Published	Promoted	Sticky	
Is One Of <All> Blog entry Chat room Page Story Video	Equals <All>	Equals <All>	Equals <All>	Submit

Bulk operations

- Choose an operation -

- Choose an operation -
- Delete
- Demote from front page
- Make sticky
- Promote to front page
- Publish
- Reload Embedded Media Data
- Remove stickiness
- Unpublish

	Type	Author	Status	Edit
☐ Blog entry	Blog entry	drupal tips	Published, Promoted	Edit
☐ taxonomy_term	Blog entry	drupal tips	Published, Promoted	Edit
☐ Drupal issue bookmark	Blog entry	drupal tips	Published, Promoted	Edit

Let's add this improved content management form to the Administration menu. If you have installed the **Administration menu module**, the new menu item will appear in the **Content** section. Go to `admin/build/menu/item/add`, and enter a **Title**, **Description**, and **Path**. Select **--- Content management** for the **Parent item** field.

Menus List Add menu **Add menu item** Settings

Title: *

The name of the menu item.

Description:

The description displayed when hovering over a menu item.

Path: *

The path this menu item links to. This can be an internal Drupal path such as `node/add` or an external URL such as `http://drupal.org`. Enter `<front>` to link to the front page.

☐ Expanded
If selected and this menu item has children, the menu will always appear expanded.

Parent item:

Weight:

Optional. In the menu, the heavier items will sink and the lighter items will be positioned nearer the top.

Recipe notes

- Note the use of a special **Node: Edit link** field in this view.

Node:
Edit Return To View
link

This, and two other administrative fields, **Node: View link** and **Node: Delete link**, come with the **Views** module. In this case we do not need the **View link** (because the node title already links to the node) or the **Delete link** (because it's one of the options in the **Options** drop-down).

Recipe 51: Views Bulk Operations— Taxonomy assignment

Ingredients

Completed Recipe 17, or applied patch at <http://drupal.org/node/199675>
 Completed Recipe 50
 Taxonomy module (core module)
 Actions module: <http://drupal.org/project/actions>

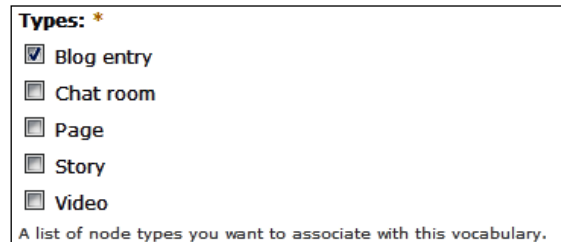
The **Views Bulk Operations** module (sometimes called VBO, for short) comes out of the box with powerful functionality as demonstrated in *Recipe 50*. Yet even more functionality is possible when combined with the **Actions** module. In this example, we enable bulk assignment of taxonomy terms for selected content items.

1. Install and enable the **Actions** and **Taxonomy** modules.
2. Create a vocabulary if you do not already have one available and populate it with terms at `admin/content/taxonomy`.

In the example below we have created a **Level** vocabulary and populated it with terms such as **Easy**, **Medium**, and **Hard**. The terms are weighted to appear alphabetically (**Hard** was given a weight of 1 while the others have a weight of 0).

Level	List	Add term
Name	Operations	
Easy	edit	
Medium	edit	
Hard	edit	

Upon adding or editing the vocabulary, you may assign the vocabulary to a content type. In this case, the **Level** vocabulary is associated with Blog entries. Your vocabulary and content may vary for this exercise.

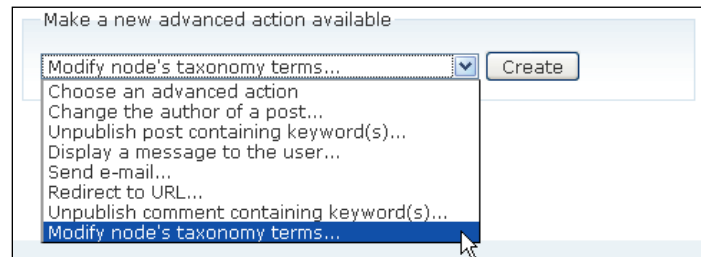


Types: *

- ☒ Blog entry
- ☐ Chat room
- ☐ Page
- ☐ Story
- ☐ Video

A list of node types you want to associate with this vocabulary.

3. Configure actions at admin/settings/actions. Select **Modify node's taxonomy terms**.

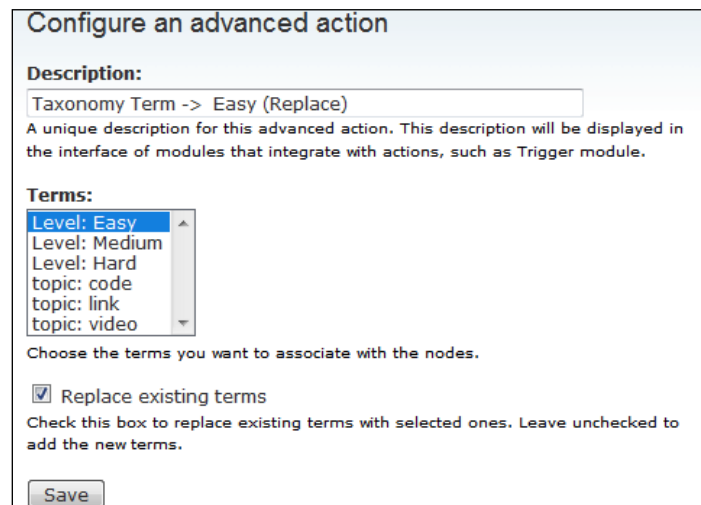


Make a new advanced action available

Modify node's taxonomy terms... Create

- Choose an advanced action
- Change the author of a post...
- Unpublish post containing keyword(s)...
- Display a message to the user...
- Send e-mail...
- Redirect to URL...
- Unpublish comment containing keyword(s)...
- Modify node's taxonomy terms...

4. Assign the **Description** and **Terms** and check the **Replace existing terms** checkbox. In order to alphabetize **Easy** and **Medium** before **Hard** we would need to add an extra space after the arrow in the description.



Configure an advanced action

Description:

Taxonomy Term -> Easy (Replace)

A unique description for this advanced action. This description will be displayed in the interface of modules that integrate with actions, such as Trigger module.

Terms:

- Level: Easy
- Level: Medium
- Level: Hard
- topic: code
- topic: link
- topic: video

Choose the terms you want to associate with the nodes.

☒ Replace existing terms

Check this box to replace existing terms with selected ones. Leave unchecked to add the new terms.

Save

Repeat Step 4 for each term you wish to bulk-assign. Use the **Description Taxonomy Term -> Hard (Replace)** without the extra space after the arrow. This is a simple sorting method, but it works. Especially since the double spaces will not display in the final form.

- Go to `admin/build/views`, and scroll down to the default views. **Add** a new view from the **admin_content** default view.

Default Views						
Default view	Title	Description	Provides	URL	Status	Actions
admin_content	Content	View, edit and delete your site's content.	Page	admin/content/node2	Overridden	Add

Make the following changes to the view.



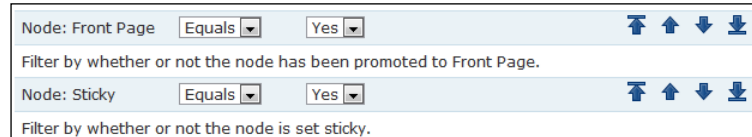
Views Quirks

An especially important button in this recipe will be the **Save and Edit** button at the bottom of the page. This button resolves a Views quirk that sometimes hides the delete filter icons even after their exposed filters have been deleted. This behavior tends to occur when cloning a view or adding a default view.

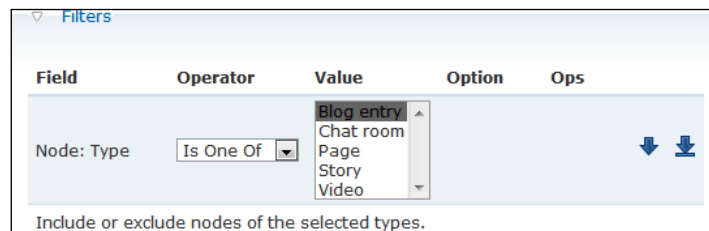
Name	admin_taxonomy
Access	admin
Description	Replace taxonomy terms
Provide Page View	Checked (Provide Page View is already checked.)
URL	admin/content/taxonomy-change
View Type	Bulk Operations View (This is already selected)
Title	Modify Taxonomy Terms
Fields	Explore, and leave as the default settings.

Filters

Note that we are not able to delete the **Front Page** and **Sticky** fields here.



This behavior is expected because we cannot remove a filter that has already been exposed. For now just select one or more node types.



We will return to the **Filters** field set after taking actions in the **Exposed Filters**.

Exposed Filters

Click on the Delete icons for both the **Node: Front Page** and **Node: Sticky** fields.

Filters

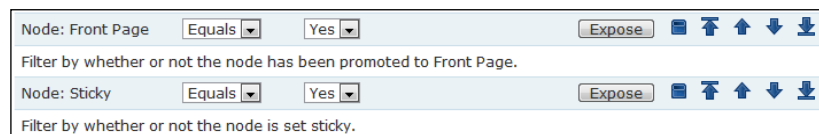
Now go back and re-open the **Filters** field set. You may be puzzled that the Delete icons may still not be available. Typically the solution, when this happens, is to press **Save and Edit** at the bottom of the page. Leave the filters settings intact until the Delete icons appear.

Save and Edit button

The **View Edit** screen remains.

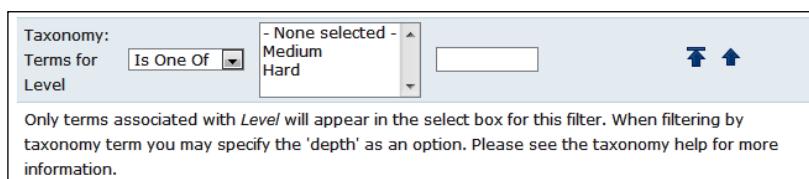
Filters

Return one last time to the **Filters** field set. Now the Delete icons should be available. Go ahead and delete the **Node: Front Page** and **Node: Sticky** fields from the filters list.



Additionally in the filters:

Add **Taxonomy: Terms for Level**, change the operator to **Is One Of** and click the **Expose** button (as usual, upon pressing the **Expose** button, the button disappears). If you find that you are missing a term be sure to apply the patch described in *Recipe 17*.



Taxonomy:
Terms for Level

Is One Of

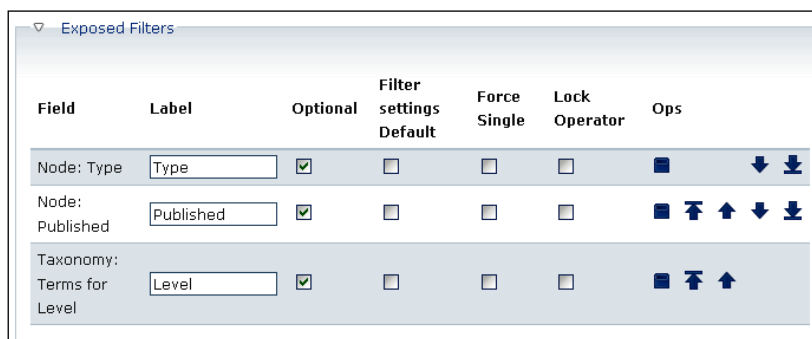
- None selected -
Medium
Hard

Expose

Only terms associated with *Level* will appear in the select box for this filter. When filtering by taxonomy term you may specify the 'depth' as an option. Please see the taxonomy help for more information.

Exposed Filters

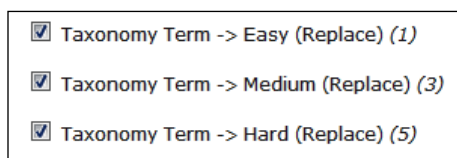
Finally, return to the **Exposed Filters** and assign a label for your vocabulary. In the example below we enter **Level** for the Level vocabulary. Enable the **Optional** checkbox.



Field	Label	Optional	Filter settings Default	Force Single	Lock Operator	Ops
Node: Type	Type	☒	☐	☐	☐	  
Node: Published	Published	☒	☐	☐	☐	    
Taxonomy: Terms for Level	Level	☒	☐	☐	☐	  

Save and Edit The **View Edit** screen remains

Click on the **Settings** tab (in the same line as **View** and **Edit**), and select the vocabulary terms operations.



☒ Taxonomy Term -> Easy (Replace) (1)

☒ Taxonomy Term -> Medium (Replace) (3)

☒ Taxonomy Term -> Hard (Replace) (5)

Test out the new view at `admin_taxonomy`. Try selecting, for instance, all entries that are not tagged with the **Medium** label, and other operations, as desired.

Type	Published	Level
Is One Of <All> Blog entry Chat room Page Story Video	Equals <All>	Is None Of <All> Easy Medium Hard

Submit

Bulk operations

- Choose an operation -
- Choose an operation -
Taxonomy Term -> Easy (Replace)
Taxonomy Term -> Medium (Replace)
Taxonomy Term -> Hard (Replace)

Execute

Type	Author	Status	Edit
taxonomy_term	drupal tips	Published, Promoted	Edit
Drupal issue bookmark	drupal tips	Published, Promoted	Edit
Function Reference for your site	drupal tips	Published, Promoted	Edit
cvs sort	drupal tips	Published	Edit

Recipe notes

- At first, all the back and forth between the **Filters**, **Exposed Filters**, and the **Save and edit** button may seem a little dizzying. The basic concepts to keep in mind:
 - Filter Delete icons are not available if the filters are exposed. The associated exposed filters must be deleted first.
 - If the Delete icon is *still* not available try doing a **Save and edit**.
- Other Views oddities may be resolved by clearing the Views cache on the **Tools** tab, `admin/build/views/tools`.
- Additional **Actions** module operations that are easily incorporated into Views Bulk Operations include changing post authors and unpublishing comments.
- If only one operation is selected for a given view, the drop-down will not appear. Instead, the button will be relabeled with the action.

- See the handbook entry on writing actions at <http://drupal.org/node/156754> and John Van Dyk's related DrupalCon presentation (video) <http://www.archive.org/details/DrupalconBoston2008-TriggersAndActionsAndHooksOhMy>.

Recipe 52: Views Custom Field

Ingredients

Views Custom Field module:

http://drupal.org/project/views_customfield

Any table, or list view, such as **swim_groups** (Recipes 3 and 4)

Relatively new and little-known, the **Views Custom Field** module can be quite handy. Use it to add row numbers, node IDs, special links, or whatever suits your needs. Once the module is installed, simply add the **Views Custom Field** and enter a label and a value.

1. Install the **Views Custom Field** module.
2. Clone the **swim_groups** view, and make the following changes:

Name	swim_groups_custom
Access	admin
URL	swim-groups-custom
Page View Type	Table View
Fields fieldset	Add Node: Title
	Add a Views Custom Field: Text field.
	Set the Label to ID and the Value to <code><?php print \$data->nid; ?></code>

Views
Custom ID
field: Text

Value:

The text that should be displayed. Include `<?php ?>` delimiters when using PHP code. Available variables:
 \$data: contains the entire record from the database (e.g. `$data->nid`).
 \$static: can be used to store reusable data per row.

Display row-specific custom text.

Click the **Move this item to the top** icon.

3. **Save** the View and go to `swim-groups-custom`. Notice the new Node ID numbers.

ID	Class
2	Water Wrigglers (Ages 2 to 3)
3	Tadpoles (Ages 3 to 5)
4	Guppies (Ages 6 to 12)
5	Flying Fish (Ages 9 - 14)
6	Adult Lessons (Ages 15+)
251	Diving (Ages 15+)

Recipe notes

- More than one custom field may be included in a single view.
- Views custom fields are certainly not limited to administrative views. However, if you *do* add data to a view that is inappropriate for users to see, be sure to limit the access for the view to the admin user.
- To add a row ID enter the following: `<?php print ++$static; ?>` (each page begins with the number 1 so it is probably best to turn off of the pager if you add a row ID).

Recipe 53: Cron setup

Cron is a Unix/Linux program used to run other programs at specific dates and times. `cron.php` is a Drupal program that many modules require to be run on a regular schedule. The following is a list of sample modules that use or require cron:

Core	Contributed
aggregator	backup_migrate
filter	calendar_ical
node	date_timezone
ping	drush_tools
search	flickrsync
statistics	location
watchdog	update_status

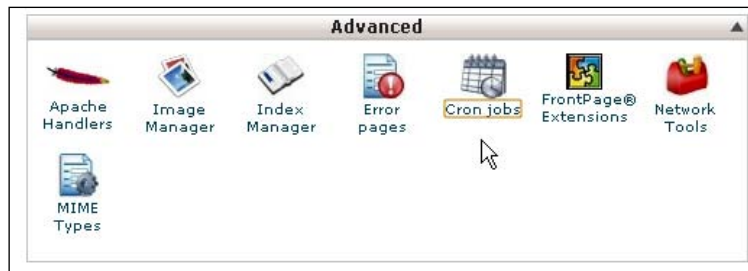
In some cases, the cron hooks that are run by `cron.php` simply clear out cache entries. In other cases, the module will not perform its primary function without cron. Regularly running Drupal's cron program is a prerequisite for *Recipe 55—Views Fast Search*.

Drupal's `cron.php` may be run manually: simply visit `http://YOURSITE.com/cron.php`, or select **Run cron** from the **Administration menu**. However, rather than visiting that URL every day—or even every half hour—it is best to configure the system's cron to do this. Alternatively you may set up a module, **Poormanscron** to run `cron.php`. Since each host system is different, with different constraints, this recipe will demonstrate three different approaches for setting up cron.

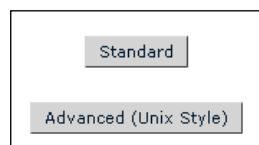
Example 1: Setting up Cron using cPanel

If you have a hosted account you may have a control panel that grants access to cron job configuration.

1. Logon to your host cPanel and click on **Cron jobs**.



2. Select (to start with) the **Standard** option.



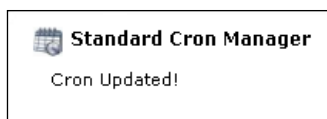
3. The **Standard** interface does not offer the option to run cron every 30 minutes. If we choose 30 minutes here, it will run cron once per hour at half-past the hour. Select **30 minutes**.

4. Enter a **command to run** such as the following:

```
/usr/bin/curl --silent --compressed http://www.example.com/  
cron.php > /dev/null 2>&1
```

The screenshot shows the 'Standard Cron Manager' interface. At the top, it says 'Entry 1'. Below that, there's a text field for 'Command to run:' containing '/usr/bin/curl --silent --compressed http://www.pingpongplaces.org/c'. Below the command field are five dropdown menus: 'Minute(s):' (showing 21, 22, 23, 24, 25, 26, 27, 28, 29, 30), 'Hour(s):' (showing Every Hour, Every Other Hour, Every Four Hours, Every Six Hours, 0 = 12 AM/Midnight), 'Months(s):' (showing Every Month, January, February, March, April), 'Day(s):' (showing Every Day, 1, 2, 3, 4), and 'Weekday(s):' (showing Every Week Day, Sunday, Monday, Tuesday, Wednesday). There is a 'Delete' button at the bottom left of the configuration area.

5. Click on the **Save Crontab** button. An update alert appears.



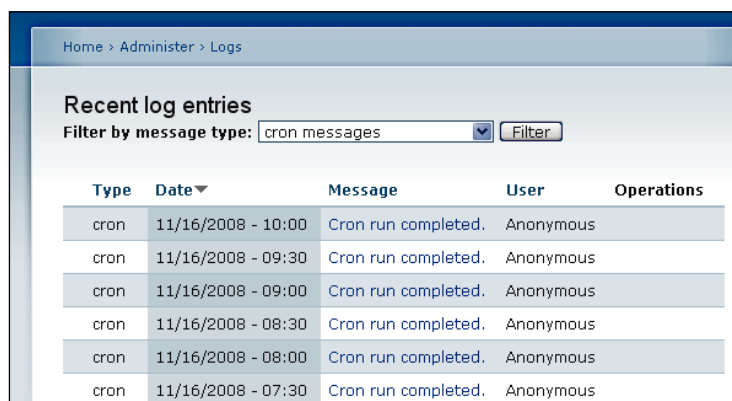
6. Click on the **Go Back** link, and check out the **Advanced (Unix Style)** settings.
7. To change the setting to run cron every half hour (rather than half-past the hour) change the **Minute** setting to ***/30**. Click on **Commit Changes**.

The screenshot shows the 'Advanced (Unix Style)' configuration interface. At the top, it says 'Advanced (Unix Style)'. Below that, there's a text field for 'Please enter an email address where the cron output will be sent:' containing 'pingpong'. Below that, there's a 'NOTE: You will receive a message every time this job runs. Using your main e-mail address is not recommended.' Below the note is a table with columns: 'Minute', 'Hour', 'Day', 'Month', 'Weekday', and 'Command'. The first row has values: Minute: */30, Hour: *, Day: *, Month: *, Weekday: *, Command: /usr/bin/curl --silent --compressed http://www.pingpong... The second row has values: Minute: 0, Hour: 0, Day: *, Month: *, Weekday: *, Command: . Below the table are buttons: 'Add Cronjob', 'Commit Changes', and 'Reset Changes'. At the bottom, there's a link: '[Go Back]'. At the very bottom, there's a footer with links: Home, Trademarks, Help, Documentation, Logout.

- Go back to the Drupal site and check the log at `admin/logs/watchdog` for cron messages. If after a half-hour you have not received a message that the **Cron run completed**, try the alternative `lynx` command.

```
/usr/bin/lynx -source http://example.com/cron.php > /dev/null 2>&1
```

If that does not work, consult your web service provider for assistance.



The screenshot shows the Drupal Administer > Logs page. Under 'Recent log entries', there is a filter dropdown set to 'cron messages' and a 'Filter' button. Below this is a table with the following data:

Type	Date	Message	User	Operations
cron	11/16/2008 - 10:00	Cron run completed.	Anonymous	
cron	11/16/2008 - 09:30	Cron run completed.	Anonymous	
cron	11/16/2008 - 09:00	Cron run completed.	Anonymous	
cron	11/16/2008 - 08:30	Cron run completed.	Anonymous	
cron	11/16/2008 - 08:00	Cron run completed.	Anonymous	
cron	11/16/2008 - 07:30	Cron run completed.	Anonymous	

Example 2: Setting up Cron on a Linux box

Many Unix and linux systems have a utility called `crontab` that enables cron setup via the `e` (for editor) switch. The commands below assume that `crontab` is configured to run with the `vim` or `vi` editor. Your system may be set for a different editor so insert, save, and exit, accordingly. As in the example above, this also assumes that the `curl` utility is located in the `/usr/bin` directory. Although they appear on separate lines below, be sure that the line beginning with `*/30` is on the same line as `http://YOURSITE.com/cron.php > /dev/null 2>&1`.

- Enter the following at the command line on your host system.

```
crontab -e
```

(Arrow down to the bottom of the file)

```
i (for insert)
```

```
*/30 * * * * /usr/bin/curl --silent --compressed http://www.  
example.com/cron.php > /dev/null 2>&1
```

(Press <Esc> key)

```
:w (for save)
```

```
:q (for exit)
```

- As in Step 8 of Setting up Cron using cPanel above, check the log entries for a completed cron run.

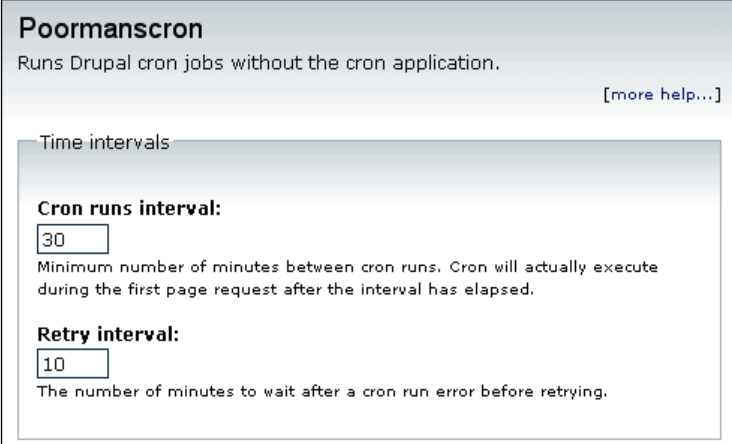
Example 3: Using the Poormanscron module

[Ingredients]

Poormanscron module: <http://drupal.org/project/poormanscron>

If you do not have access to the cron utility on your host system install Poormanscron. The module invokes the cron hooks of all enabled modules at periodic intervals.

1. Install and enable the **Poormanscron** module.
2. Configure the module at `admin/settings/poormanscron`. Set the **Cron runs interval** to 30 minutes.



The screenshot shows the configuration page for the Poormanscron module. At the top, it says "Poormanscron" and "Runs Drupal cron jobs without the cron application." with a "[more help...]" link. Below this is a section titled "Time intervals". Inside this section, there are two settings: "Cron runs interval:" with a text input field containing "30", and "Retry interval:" with a text input field containing "10". Below the "Cron runs interval" field, there is a description: "Minimum number of minutes between cron runs. Cron will actually execute during the first page request after the interval has elapsed." Below the "Retry interval" field, there is a description: "The number of minutes to wait after a cron run error before retrying."

3. Check the log entries for a completed cron run, as in *Example 1*, Step 8.

Recipe notes

- Where did the **curl -silent -compressed** command come from? Check the **scripts** directory of your Drupal installation. There are two sample scripts for running `cron.php` either via the `curl` or `lynx`. Use whichever command you have access to on your system. Although it's not included in the `curl` script, we have added `>/dev/null 2>%1`, which will prevent emails from accruing every time cron is run.

- More information about the `/dev/null` redirect may be found at <http://www.xaprb.com/blog/2006/06/06/what-does-devnull-21-mean/> and <http://tinyurl.com/5ptffz>.
- To determine the location of the command on your host system (not necessary, if using **Poormanscron**) consider entering one the following on the command line:
 - `which curl`
 - `which wget`
 - `which lynx`

Recipe 54: Formatting Views queries

Ingredients

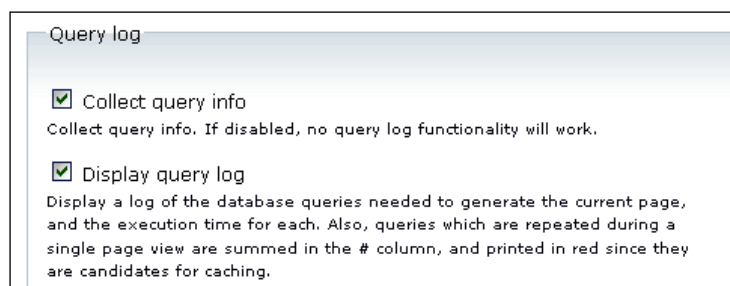
Devel module: <http://drupal.org/project/devel>

Instant SQL Formatter: <http://www.dpriver.com/pp/sqlformat.htm>

Chrome Browser (Optional): <http://www.google.com/chrome>

The Views module constructs a query to select and sort specific data from a Drupal site. The Views UI greatly eases the task of writing **Structured Query Language (SQL)**. This recipe enables the display of the Views query along with a utility for "pretty-printing" for easier reading.

1. Install the **Devel** module.
2. Go to the **Devel** module settings at `admin/settings/devel` and enable the collection and display of query information.



3. Open a page with a view. Find **views_build_view** in a list of functions at the bottom of the page. The **Devel** module is displaying the full query and the time that it took to run in milliseconds. Consider using Google's **Chrome browser** for this text search. Chrome helpfully displays markers on the browser scrollbar signaling where all the search results are located on the page.

A sample Views-building query may look like this:

```
SELECT node.nid, node.created AS node_created_created, node.title
AS node_title, node.changed AS node_changed FROM node node LEFT
JOIN content_type_media_hit node_data_field_referenced_press_
release ON node.vid = node_data_field_referenced_press_release.
vid WHERE (node.type IN ('media_hit')) AND (node_data_field_
referenced_press_release.field_referenced_press_release_nid = 233)
ORDER BY node_created_created DESC LIMIT 0, 1
```

4. The query will be easier to read upon reformatting. Open a browser tab for the Instant SQL formatter at <http://www.dpriver.com/pp/sqlformat.htm>.
5. Set the database to **MySQL** and the **Output** to **SQL(Text)**.

Choose a database	MySQL	Output:	SQL(Text)
-------------------	-------	---------	-----------

Additionally, you may set options such as the case for keywords and the placement of line breaks.

Keywords case:	Uppercase
Identifier case:	Lowercase
Function case:	InitCap
Linebreaks with comma:	
☒ After	
☐ Before	
☐ Before with space	
List and Parameters Style:	
☒ Stacked	
☐ Not Stacked	

6. Click on the **Format SQL** button, and the text will appear below in a newly readable form.

```

/* Powered by General SQL Parser (www.sqlparser.com) */
SELECT  node.nid,
        node.created AS node_created_created,
        node.title   AS node_title,
        node.changed AS node_changed
FROM    node node
        LEFT JOIN content_type_media_hit node_data_field_
referenced_press_release
        ON node.vid = node_data_field_referenced_press_release.
vid
WHERE    (node.TYPE IN ('media_hit'))
        AND (node_data_field_referenced_press_release.field_
referenced_press_release_nid = 233)
ORDER BY node_created_created DESC
LIMIT   0,1

```

Recipe notes

- In the Devel settings you may wish to temporarily check **Store executed queries**.
- Once the queries are stored this database query will display all the queries related to views.

```
SELECT function, query FROM `devel_queries` WHERE function like
"%views%"
```

Uncheck the query storage when you are done examining queries.

- The **Drupal For Firebug** module, <http://drupal.org/project/drupalforfirebug>, will also display Drupal queries along with additional debugging information for users with the **Access Firebug Debug** permission.
- If you notice that some Views queries take an especially long time to run (slow-to-run query times display in red in Devel) consider adding an index to selected fields in your database to increase the query's efficiency. CCK tables (content_type_CONTENTNAME, or content_field_FIELDNAME for fields with multiple values) are, by default, already indexed by node ID.
- Consider enabling the Slow Queries log in MySQL <http://dev.mysql.com/doc/refman/5.0/en/slow-query-log.html>.
- Additional alternatives for "pretty-printing" SQL queries include the free PSPad text editor for Windows users and a desktop version of the **dpriver** software above. Download PSPad at pspad.com. Additionally, download and unzip the SQL Format extension <http://pspad.com/files/pspad/rozsireni/133-SQLFormat.zip> to the PSPad/Script/JScript directory. Choose **Scripts | Recompile scripts**, and then **Scripts | Format Code | SQL Format w/ Indent**.

Summary

This chapter focused on tools and views for site administrators and editors.

Resources include:

- The Administration menu, including links to views and development functions
- Module information links
- Reduced complexity for the Views User Interface, either by sorting drop-down lists, or removing options for the editor role
- Editable fields
- Views Bulk Operations
- Cron setup
- Automated formatting of SQL statements for easier reading

6

Views Galore

This chapter offers an abundance of interesting Views. Half of the Views Galore recipes stand alone, while the other half work in pairs. The pairs include Recipes 59 and 60 (Photo gallery with Lightbox2 and jQuery), 61 and 62 (Google Maps and Proximity Search), and 65 and 66 (Flags for bookmarks, RSVPs, and other use cases).

Recipes 57 and 58 are probably the only two "quick" ones here. For the others, give yourself some time and focus, assemble all the ingredients, let them simmer a bit, and enjoy the results.

Recipe 55: Views Fusion

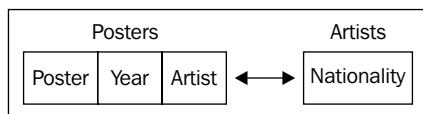
Ingredients

Recipe 31, Steps 1–6

Views Fusion module: http://drupal.org/project/views_fusion

Node Reference Patch: <http://drupal.org/node/123482>

In *Recipe 31* (Chapter 3) we created two content types: **Artist** and **Poster**. Each artist may produce zero, one, or more posters. We used a Node Reference field in the Poster content type – populated by a simple view – to associate an artist with each poster. Finally, we created a `node-artist.tpl.php` template file, which displayed specific artist information, along with their posters. In *Recipe 55*, we also display the two content types at once, but the approach is quite different – with the **Views Fusion** module, we can actually combine two views into one. Essentially, we will be creating, and then joining the posters and artists views.



Patch the Node Reference module

1. At the time this recipe is being written, a patch is required to the **Node Reference** module. A few caveats apply. First, check to see if the patch has already been applied. Go to <http://drupal.org/node/123482> and note the **Status** in the information block.

fago - February 28, 2007 - 08:55	
Project:	Content Construction Kit (CCK)
Version:	5.x-1.x-dev
Component:	nodereference.module
Category:	feature request
Priority:	normal
Assigned:	fago
Status:	needs review

If the status has changed to **Fixed**, you won't have to perform this patch, and you may skip to Step 4.

2. If the patch must be performed, be sure that the patch version is appropriate for your version of CCK. Determine the CCK version at `admin/build/modules`, or by reading the `nodereference.info` file in the **cck** directory. The CCK Views Fusion Reference 1.9 patch applies both to CCK versions 5.x-1.9 and 5.x-1.10.
3. Go to the host directory and perform the following commands:

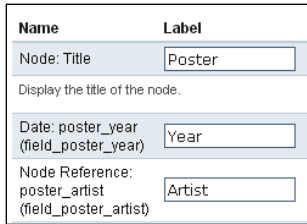
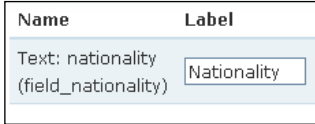
```
cd <DRUPALROOT>/sites/all/modules/cck
cp nodereference.module nodereference.module.orig
wget http://cdn1.drupal.org/files/issues/cck_views_fusion_reference_1.9.patch
patch < cck_views_fusion_reference_1.9.patch
```

With these steps we go to the `cck` directory, create a backup of the `nodereference.module` file, and download, and perform the latest patch.

Views setup

4. Create two simple views, **posters**, and **artists**.

Name	posters	artists_nationalities
Description	List of Posters and Production Years	List of nationalities
Provide Page View	Checked	Checked

URL	posters	artist-nationalities
View Type	Table View	Table View
Title	Posters	Artists
Use Pager	Checked	Checked
Nodes Per Page	50	50
Fields	Node Title – Option: As link (Set the Label to Poster.) Date: poster_year (field_poster_year) (Set the Label to Year.) (This field is optional, add if the poster_year field is available as part of the content type.) Node Reference: poster_artist (field_poster_artist) 	Text: nationality (field_nationality). (Set the label to Nationality) 
Filters	Node: Published Equals Yes Node: Type Is One Of Poster	Node: Published Equals Yes Node: Type Is One Of Artist
Sort Criteria	Text: Last name (field_artist_last_name), Descending	None

Views Fusion installation and configuration

5. Install and enable the **Views Fusion** module.
6. Visit `admin/build/views` (a rather familiar URL by now) and note the new **Fusion** tab. Click on the **Fusion** tab, and configure, as follows:

primary view	posters
fuse with view	artist_nationalities
using	Node Reference: poster_artist (field_poster_artist) (This option will be available if you have patched the node reference module, and created the node reference field.)

After assigning these settings, click on the **Add fusion** button.

Add another views fusion

primary view: *
posters

fuse with view: *
nationality

using: *
 Node Reference: poster_artist (field_poster_artist)
 Node Reference: poster_artist (field_poster_artist)
 Node Reference: poster_artist (field_poster_artist) - reverse direction

A list of settings is created at the top of the **Fusion** page. Note that listed items may be deleted, but not edited.

Views List Add **Fusion** Import Tools

This screen allows the fusion of two views. Select the main view you want to use and then select a second view you would like to "fuse" with the first view. Fields, filters and sorts from the fused view will be intelligently merged into the primary view. [\[more help...\]](#)

primary view	fused view	using
posters	nationality	Node Reference: poster_artist (field_poster_artist) delete

7. Open the posters view at <http://YOURSITE.com/posters>.

Poster	Year	Artist	Nationality
All Arms	1976	Tawfiq Abdel Al	Palestinian Territory, Occupied
Don't Say You Didn't Know	1982	Iris Dishon	Israel
Edward Said in Berkeley	2006	Jos Sances	United States of America
Israel/40th Anniversary	1988	Iris Dishon	Israel
Unarmed Truth	2004	Jos Sances	United States of America

The two views are joined. Note that artists who do not have associated posters do not appear in this view.

Recipe notes

- Views Fusion documentation appears in the Views handbook at <http://drupal.org/node/130944>.
- Be sure also to read the module's `README.txt` file.
- When setting up Views Fusion, be sure to clear the views cache often `admin/build/views/tools`.

Recipe 56: Views Fast Search

Ingredients

Completed Recipe 53 (cron setup)

Search module, part of Drupal core

Views Fast Search: http://drupal.org/project/views_fastsearch

Views Fast Search merges the Views and Search modules in a manner that many will find superior to the default search functionality. **Views Fast Search** exposes a keyword search filter to Views. Additional exposed filters may also be included in the view.

Setup

1. Ensure that cron is being run regularly, by completing one of the three options in *Recipe 53*. Among other things, this step will update the search index.
2. Install the **Views Fast Search** module.
3. Enable the core **Search** module and the **Views Fast Search** module

Using the Default View

4. Go to the **Default Views** section of `admin/build/views`. Add the **views_fastsearch** default view.

views_fastsearch	Search	Search	Page	search/fast	Enabled	Add Disable
------------------	--------	--------	------	-------------	---------	--

5. Explore the view. Note the View Type of **Search Results**. The field and sort are both set to **Search: Score**. The filter is set to **Search: Fast Index** with an **AND** operator and it is exposed and locked. No changes are required in this step.

6. **Save** the view, and try it out at <http://YOURSITE.com/search/fast>.



Using an Alternative Imported View

7. For an alternative view—to see additional possibilities with **Views** **Fast Search**—go to `admin/build/views/import`, and import the view found at http://drupal.org/files/issues/search_view.inc_.txt. Be sure to remove the first line containing the PHP tag before pressing the **Submit** button.

Change the URL to **search/alt** before saving the view.

8. Visit the search URL, at <http://YOURSITE.com/search/alt>.



This view, while just as easy to produce, has a very different look.

9. Let's go back to the view setup at <http://YOURSITE.com/search/alt/edit>, and review some of the properties of our imported view. Like our first Fast Search view, the search includes the exposed filter **Search: Fast Index** with an **AND** operator. But this is where the similarities end. The View type is a regular **Table View**, and the Sort is by date. Also note the special **Page Empty Text** as follows:

```
Note the <?php
if ($_GET['filter0'])
{ print 'No matching entries were found. ';
}
?>
```

This ensures that the table view will not say that "No matching entries were found" unless the URL includes the `filter0` parameter (thus, the message will not show when the `search/edit` view is displayed the first time).

10. Optionally, you may edit the **Search: Fast Index Operator**, changing it from **AND** to **AND (Empty all)**. Now if the user does not enter search text, (a search by Content type will still yield results.

Replacing the Search block

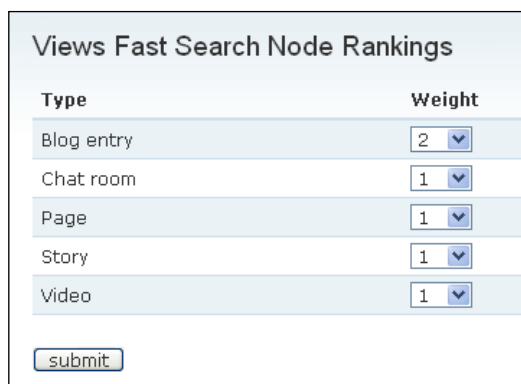
11. You may wish to replace the standard search block so that it runs the Views Fast Search. Create a new block at `admin/build/block/add`, with the following content:

```
<form id="views-filters" method="get" action="/search/fast">
  <div class="form-item">
    <input type="text" class="form-text" size="10"
      id="edit-filter0" name="filter0" maxlength="255"/>
    <input type="submit" class="form-submit" value="Search"
      id="edit-submit"/>
  </div>
</form>
```

Be sure to set the input format to **Full HTML** and replace the `action` with the URL of your particular view. Additional approaches to creating a **Views Fast Search** block are discussed at <http://drupal.org/node/189531>.

Recipe notes

- Install the **Views Fast Search - Node type rankings** module, which comes with **Views Fast Search**, to prioritize selected content types in the search results. Configure this module at `admin/settings/vfs_ranking_nodetype` (this module may slow down your search).



The screenshot shows a web form titled "Views Fast Search Node Rankings". It contains a table with two columns: "Type" and "Weight". The table lists five content types: "Blog entry", "Chat room", "Page", "Story", and "Video". Each type has a corresponding weight value in a dropdown menu: "Blog entry" is 2, and the others are 1. Below the table is a "submit" button.

Type	Weight
Blog entry	2
Chat room	1
Page	1
Story	1
Video	1

- Install the **Porter-Stemmer** module to be able to find a word such as "blogging" when searching for "blog" from `http://drupal.org/project/porterstemmer`.
- Additional resources:
 - Case sensitivity and partial word search in Views Fast Search: `http://drupal.org/node/201530`.
 - Result Count and Result Range for Views Fast Search: `http://drupal.org/node/338634`.
 - The Lullabot company put together an excellent article on the Views Fast Search module: `http://www.lullabot.com/articles/custom_search_forms_views_and_fastsearch`.
 - Solutions offered for issues with search: `http://www.darcynorman.net/2006/06/07/drupal-search-funkiness/`.

Recipe 57: YouTube Video Bar

[**Ingredients**]
An existing view (this recipe is easy to adapt to a new view, as well)

In this recipe, YouTube and Views do all the heavy lifting, while our work is easy. A minute's effort will place a bar of four pop-up YouTube videos at the top of a view.

1. Visit the **Google Video Bar Wizard** at <http://www.google.com/uds/solutions/wizards/videobar.html>.

Google AJAX Search API (Beta)

[Google Code Home](#) > [Google AJAX Search API](#) > [Wizards](#) > **Video Bar**

Put YouTube Videos on Your Web Page

Embed a strip of YouTube videos on your on your web page and let your users watch the video channels you've selected without leaving your site. Customize how the video bar should be displayed, and this wizard will write the code for you.

1 Customize it

Orientation: ☐ vertical ☒ horizontal

Popular Channels: ☐ Most Viewed Videos ☐ Top Rated Videos ☐ Recently Featured Videos

YouTube Channels:
Note: You can either specify a single channel (e.g., lonelygirl15) or a comma separated list of channels

Search Expressions:
Note: You can specify a single search expression (e.g., hamster dance) or a comma separated list of expressions




powered by 

2 Add code to your site

Copy and paste the following where you want your video bar to appear. Do not place it within the `<head> ... </head>` section of your page unless you plan on relocating the `<div id="videoBar-bar"></div>` element out of this chunk of code.

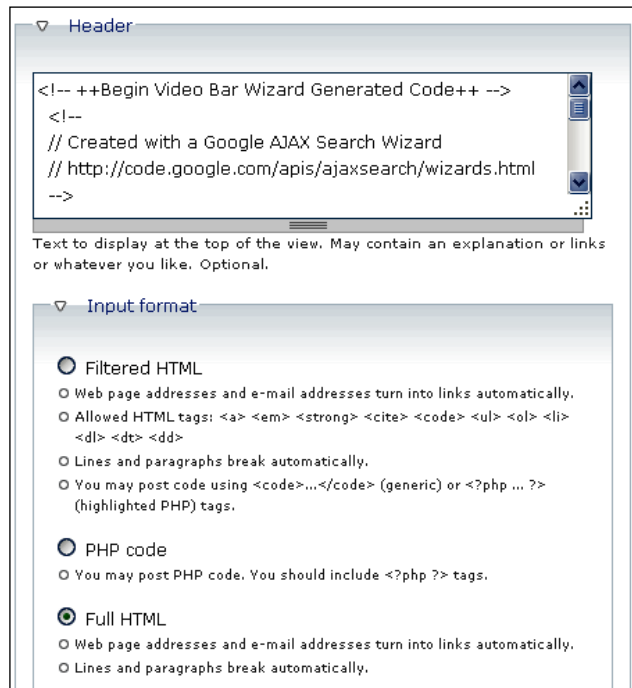
```

<!-- ++Begin Video Bar Wizard Generated Code++ -->
<!--
// Created with a Google AJAX Search Wizard
// http://code.google.com/apis/ajaxsearch/wizards.html
-->

<!--
// The Following div element will end up holding the actual videobar.
// You can place this anywhere on your page.

```

2. Unselect **Most Viewed Videos**. Replace the **YouTube Channels** with the channel of your choice (multiple channels are separated by a comma).
3. Copy the code from the wizard into the header of an existing page view. Be sure to change the input type to **Full HTML**.



4. Save the View, and visit its URL. Enjoy the new videos.

Recipe notes

- With a bit of JavaScript editing, you may add more than one video bar on the top of a view. Copy the code block, and modify as follows:
 - After it appears the first time, remove subsequent appearances of the Ajax Search API script and stylesheet (this section of code is marked with a comment that "If you are already using the AJAX Search API, then do not include it or its stylesheet again").
 - Make sure the div IDs differ, for instance, change the second div to `<div id="videoBar-bar2">`

- Change the videobar assignment to the new ID, for instance, **new SvideoBar(document.getElementById("videoBar-bar2")),**
- Change the **executeList** to the new channel or search term
- This recipe may be interpreted more broadly. Try creating a vertical bar and placing it in a block. Or, try an entirely different script. Quick and easy tools available at Google include the Map Search Wizard, the Video Search Wizard, and News, Book, and Blog Bar Wizards. These are available at <http://code.google.com/apis/ajaxsearch/wizards.html>.
- Other excellent approaches to including videos inside a view include the Embedded Media Field module <http://drupal.org/project/emfield/> and Kaltura <http://drupal.org/project/kaltura>.



Recipe 58: Views Bonus Pack Export

Ingredients

Views Bonus Pack module: http://drupal.org/project/views_bonus
 Bonus Views Export module (comes with Views Bonus Pack, above)

The **Views Bonus Pack** comes with a number of style plugins. They are listed in the *Views Style Plugins, Appendix C*. One **Views Bonus Pack** module that is definitely worth incorporating into common practice is Views Export. The sample view below assumes that the location module is installed. However, any fields available to Views may be selected for export.

1. Install the **Views Bonus Pack** module, and enable the **Bonus: Views Export** module.
2. Create a Page View such as the following:

Name	export_addresses
Description	Export address to CSV
Provide Page View	Checked
URL	export-addresses
View Type	Views Bonus: CSV Export
Title	(The Title is ignored in the export.)
Fields	Select all the fields you wish to export to a comma-delimited file.

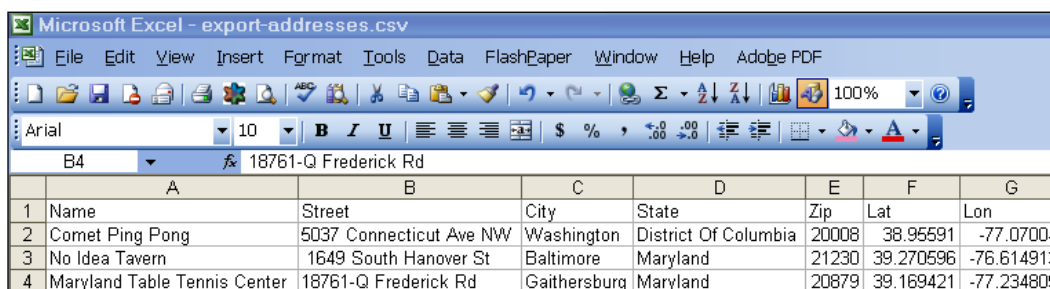
Fields

Name	Label	Handler	Option	Sortable	Default Sort
Location: Name	Name			No	None
Location: Street	Street			No	None
Location: City	City			No	None
Location: Province name	State			No	None
Location: Postal Code	Zip			No	None
Location: Latitude	Lat			No	None
Location: Longitude	Lon			No	None

Filters	Node: Published Equals Yes
	Node: Type Is One Of Places (or the content type of your choice).

3. Save the View, and visit the associated URL.

Upon visiting the URL, the browser will download the `export-addresses.csv` file. Depending on how your browser is configured Excel may open right away, or you may need to manually open the file. Note that the labels from Views become the column headers.



	A	B	C	D	E	F	G
1	Name	Street	City	State	Zip	Lat	Lon
2	Comet Ping Pong	5037 Connecticut Ave NW	Washington	District Of Columbia	20008	38.95591	-77.07004
3	No Idea Tavern	1649 South Hanover St	Baltimore	Maryland	21230	39.270596	-76.614913
4	Maryland Table Tennis Center	18761-Q Frederick Rd	Gaithersburg	Maryland	20879	39.169421	-77.234809

Recipe notes

- As an alternative to selecting the view type of **Views Bonus: CSV Export**, you may select a standard view type, such as **Full Node**, and then add the argument of **CSV: CSV File Selector** (with a Default value of **Display All Values**). Change the URL accordingly. For instance, if the URL were set to `places`, the following URLs would apply: `http://www.example.com/places` (to show full nodes) and `http://www.example.com/places/csv` (to create the Comma Separated Values file). Oddly enough, even if you select the CSV filter, the following URLs will also work, and will open in their respective associated applications.
`http://www.example.com/places/doc`
`http://www.example.com/places/txt`
- If you have a very large dataset that causes PHP to run out of memory when performing the export, consider using the **Views: Many Node Export** module, http://drupal.org/project/many_node_export_views.

Recipe 59: Photo Gallery

Ingredients

Content module: <http://drupal.org/project/cck>

ImageAPI module <http://drupal.org/project/imageapi>

ImageAPI GD2 module, comes with ImageAPI module

ImageField module: <http://drupal.org/project/imagefield>

ImageCache module: <http://drupal.org/project/imagecache>

ImageCache UI module, comes with Imagecache module

Views Bonus Pack module: http://drupal.org/project/views_bonus

Bonus Grid View module, comes with Views Bonus Pack

There are many ways to present images on a website. In this approach we create a gallery using a collection of Views-enabled modules. We will enhance the gallery in *Recipe 60* and *83*.

1. Install and enable each of the modules listed in the Ingredients section.

Set up the Photo Content type

2. Create a new content type named **Photo** at `admin/content/types/add`.

Name	Photo
Type	photo
Title Field Label	Title
Body Field Label	<i>(empty)</i>
Default Options	Published (Uncheck Promoted to front page.)
Default Comment Setting	Disabled
Attachments (Present if the upload module is enabled)	Disabled

Click the **Save content type** button.

3. Edit the Photo content type at `admin/content/types/photo/fields`, and choose **Add Field** to add a field for photos, as follows:

Name	photo_photo (we're using the convention of <code><CONTENTTYPE>_<FIELDNAME></code> in naming the field.)
Field Type	Image: Image (Click the Create Field button)
Label	Photo
Image Path	photos
Enable custom alternate text	Check
Help Text	Upload a jpg, jpeg, png, or gif file.

Click on the **Save Field Settings** button.

Configure ImageCache

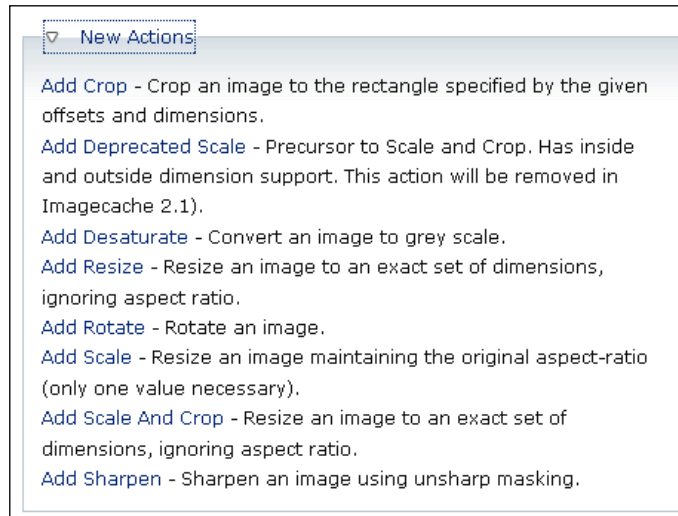
4. Begin configuration of **ImageCache**, at `admin/build/imagecache/add`. Set the **Preset Namespace** to **130x130square**, and click the **Create New Preset** button.



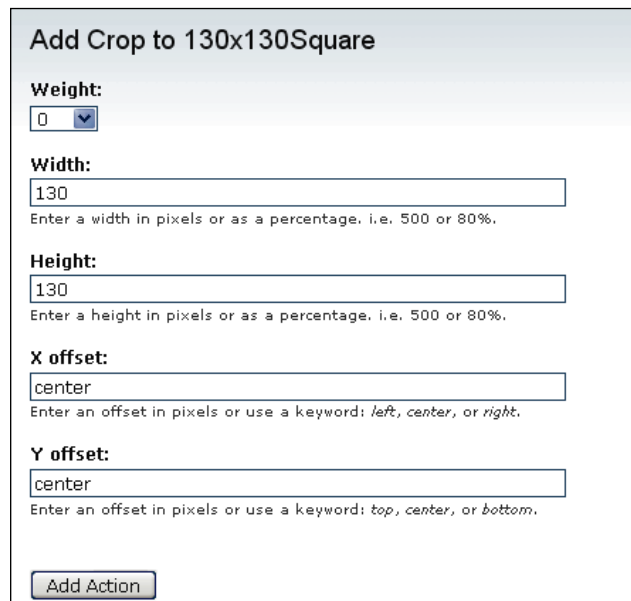
Files permissions

If an error message crops up, be sure that you have a **files** directory, and that it is accessible. After creating a **files** directory, many people run the `chmod 777 files` command to open the permissions. This not a security best practice. A better approach is to determine the name of the apache user on your server, and `chown APACHEUSERNAME files` (this command changes the ownership of the **files** directory to the apache user). Run the `ps ax` command to help determine the Apache username. Common Apache names from various systems include **apache**, **httpd**, **nobody**, **www-data**, and **wwwrun**.

- Expand the list of **New Actions**, and choose **Add Crop**.



- Set the Width and Height to **130** and set the X and Y offsets to **center**. Click the **Add Action** Button.

A screenshot of a dialog box titled 'Add Crop to 130x130Square'. The dialog box contains several input fields and a button. The 'Weight' field is a dropdown menu set to '0'. The 'Width' field is a text input containing '130', with a note below it: 'Enter a width in pixels or as a percentage. i.e. 500 or 80%'. The 'Height' field is a text input containing '130', with a note below it: 'Enter a height in pixels or as a percentage. i.e. 500 or 80%'. The 'X offset' field is a text input containing 'center', with a note below it: 'Enter an offset in pixels or use a keyword: left, center, or right.'. The 'Y offset' field is a text input containing 'center', with a note below it: 'Enter an offset in pixels or use a keyword: top, center, or bottom.'. At the bottom of the dialog box is a button labeled 'Add Action'.

Add Photos

7. Add photos at `node/add/photo`. Add at least 8 photos (the **Node Go To** module that we saw in *Recipe 34*, may come in handy here, as well, to present a new node edit form after each form submission).

Home » Create content

Submit Photo

Title:

photo_photo

Images are not saved until the form is submitted.



Filename: brisbane_botanical_gardens_022.jpg ☐ Delete

Alternate text:

Alternate text to be displayed if the image cannot be displayed.

View setup

8. Create a **photos** view.

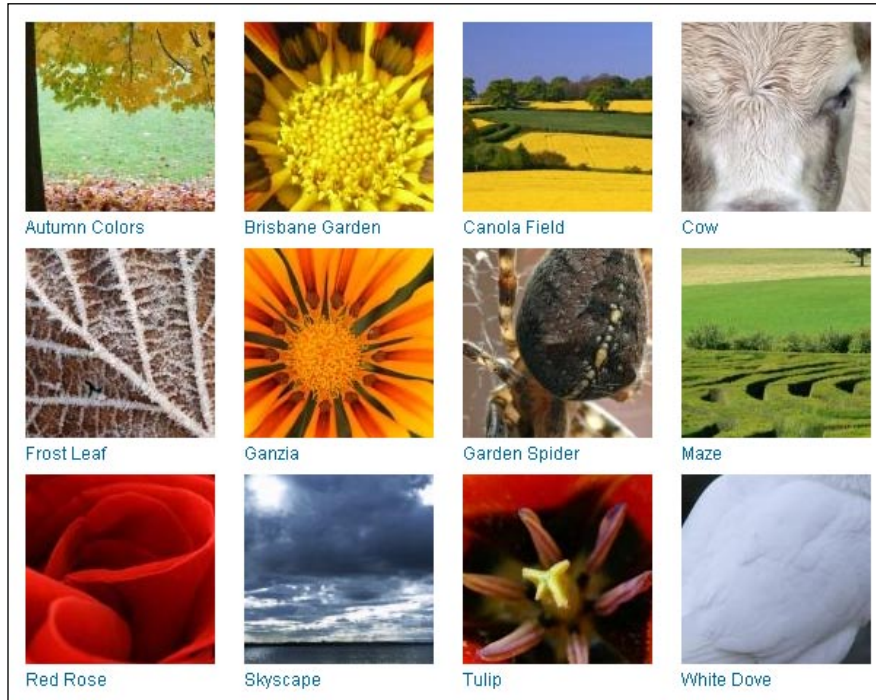
Name	photos
Description	Gallery of Photographs
Provide Page View	Checked
URL	photos
View Type	Bonus: Grid View
Title	Photos
Use Pager	Checked
Nodes Per Page	24
Fields	Image: photo_photo (field_photo_photo), Select the 130x130Square Option.

Node Title Option: As link

Name	Label	Handler	Option
Image: photo_photo (field_photo_photo)		Do not group multiple values ▾	130x130Square ▾
Node: Title		Normal ▾	As link ▾
Display the title of the node.			

Filters	Node: Published Equals Yes
	Node: Type Is One Of Photo
Sort Criteria	Node: Title Ascending

9. **Save** the view, and visit the **photos** page. Enjoy the View!



Recipe notes

- To change the grid column count to three, enter the following in the argument code section of the photo view (the Argument Handling Code Fieldset is contained within the Arguments fieldset).

\$view->gridcount = 3;

Do not include the opening or closing PHP tags.

- Several releases of the **ImageField** module introduced an odd bug preventing the display of images. An effective work-around is described in comment #1 at <http://drupal.org/node/336452>, but the best fix is to upgrade to the current version of **ImageField**.

Recipe 60: Gallery with Lightbox2 and jQuery Update

Completed Gallery, Recipe 59
jQuery Update: http://drupal.org/project/jquery_update
Lightbox2 module: <http://drupal.org/project/lightbox2>

Lightbox2 enhances the gallery, presenting a larger version of the thumbnail images along with navigation arrows. The module requires the **jQuery Update** module.

Installing jQuery Update

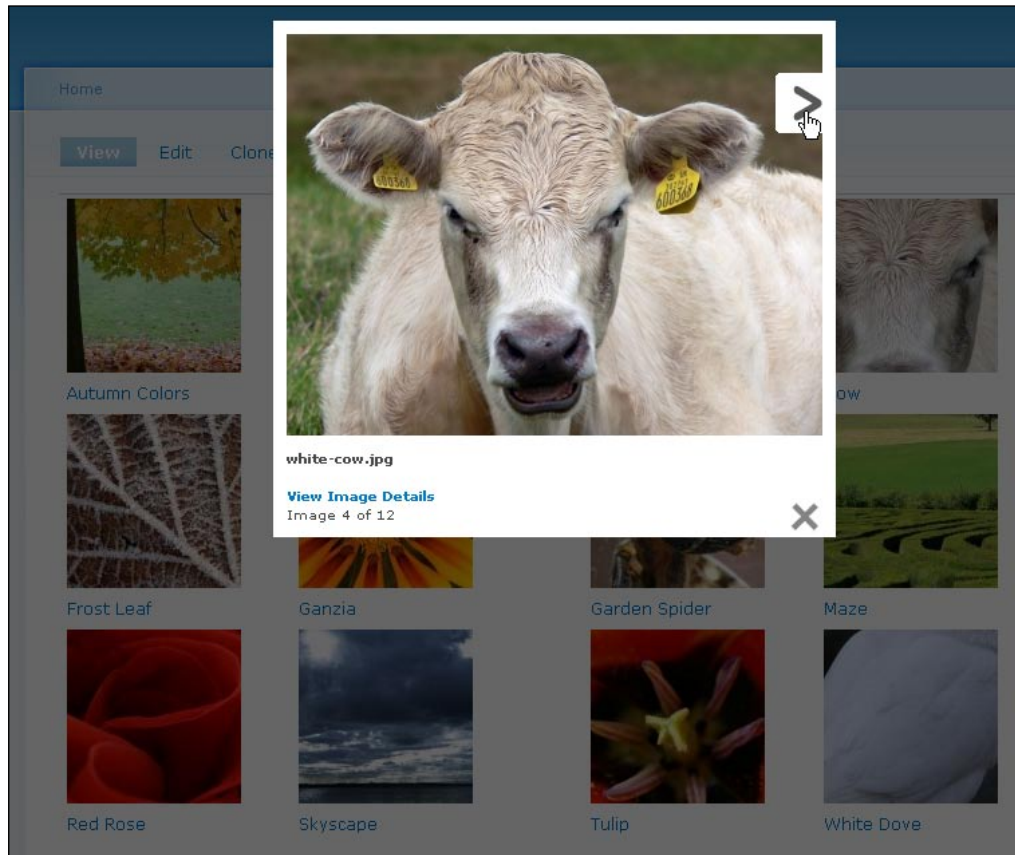
1. Install and enable the **jQuery Update** module.
2. Back up the JavaScript files from the Drupal site's `misc` directory and overwrite them with the new JavaScript files found in the **jQuery Update** module. Be sure to substitute the appropriate pathname for `<DRUPALROOT>` in the steps below.

```
mkdir /backups/  
mkdir /backups/misc  
cd <DRUPALROOT>/misc  
cp *.js /backups/misc  
cd <DRUPALROOT>/sites/all/modules/jquery_update/misc  
cp *.js <DRUPALROOT>/misc/
```

Installing Lightbox2

3. Install and enable the **Lightbox2** module.
4. Open up the photos view, go to the **Fields** fieldset, and change the **Image Option** to **Lightbox2: 130x130Square->Original** (if the option is not present in the drop-down list, try clearing the cache. This is especially easy to do with the administration menu *Installed, Recipe 45*).

5. **Save** and open the **photos** view. Note that a single click on the image reveals the full photo. Mousing over the Lightbox2 photo exposes arrows that may be used to navigate the rest of the gallery.



Recipe notes

- Lightbox2 configuration is not required for this recipe. However, it is definitely worth reviewing the extensive options available at [admin/settings/lightbox2](http://drupal.org/node/249827).
- Be sure to become familiar with the keyboard shortcuts for rapid navigation <http://drupal.org/node/249827>.
- For a comparison of lightbox-style modules (prepared by the author of the Drupal **Lightbox2** module) see <http://drupal.org/node/266126>.

Recipe 61: Google Maps

Ingredients

jQuery Update: http://drupal.org/project/jquery_update
Country Codes API: http://drupal.org/project/countries_api
Google Maps Tools: <http://drupal.org/project/gmaps>
Gmaps Content Fields, Gmaps Views, and and Google Maps. (These modules come with Google Maps Tools, above.)
Clean URLs enabled
Complete Recipe 60, Steps 1 and 2

There are a number of different mapping solutions available to Drupal users. In this recipe we use a relatively new module called **Google Maps Tools**.

The module is rich in functionality, offering a selection of five different CCK field types. For our map of trailhead parking sites, we will use the **Map marker** field. We'll assign each trailhead its geographic coordinates by viewing aerial photography and manually placing a marker.

The assignment of geographic coordinates (generally, latitude and longitude) is known as "geocoding".

Recipe 62 is also based on **Google Maps Tools**, so try out both to get a sense of the extent of this module.

Setup Google Maps Tools

1. Install and enable the modules listed in the ingredients. Instructions for installing jQuery Update are found in Steps 1 and 2 in *Recipe 60*.
2. Ensure that **Clean URLs** are enabled at `admin/settings/clean-urls`.
3. Visit <http://code.google.com/apis/maps/signup.html> to get a Google Maps API key. Enter the domain name of your site, and check that you have read and agree to Google's terms and conditions. Press the **Generate API Key** button, and copy the key to the clipboard.
4. Paste the Google Maps Key into the Google Maps Tools configuration page on your Drupal site at `admin/settings/gmaps`. Leave the rest of the gmaps settings intact, and click the **Save configuration** button.

5. Add a **Hybrid Map** configuration at `admin/build/gmaps-map-config/add`. (**Hybrid** is Google's term for an overlay of both street and satellite views. The hybrid map will help us to better locate trailhead parking.)

Configure the settings as follows:

Map type	Earth Map
Configuration name	Hybrid Map
Default Zoom level	6
	Note: Zoom levels range from 0 (Satellite view, or no zoom) to 19 (close-up).
Allowed Tiles	Select Normal , Satellite , and Hybrid maps Deselect Physical Map
Default Tile	Hybrid map
Methods	Enable dragging Enable automatic center and zoom Enable height resize Enable info window
Map Control	Large Map Control
Map Type Control	Standard

The resulting summary page at `admin/build/gmaps-map-config` looks like this:

Map configurations		List	Add map configuration	
Name ▲	Type	Allowed tiles	Operations	
Hybrid Map	Earth map	☐ Normal map ☐ Satellite map ☐ Hybrid map	edit	delete

Create and populate the Content types

6. Create a **Trailhead Parking** content type at `admin/content/types/add`, with the following settings:

Name	Trailhead Parking
Type	trailhead_parking
Description	Parking site for access to walking and biking trails
Body Field	Omit field
Promoted to front page	Uncheck

7. Add a **Map marker** field to **Trailhead Parking** at `admin/content/types/trailhead-parking/add_field`.

Address
☐ Address Field

Geocoded address
☐ Geocoded Address Field

Geocoded address marker
☐ Geocoded Address Marker Field

Map point
☐ Point Field

Map marker
☒ Marker Field

8. Leave most of the field settings as the default, with the following edits:

Name	th_parking_location
Label	Location
Point settings: Map Configuration	Hybrid Map <i>(This may be preset, if it's your only map configuration.)</i>

9. Populate the **Trailhead Parking** with content. Pan the map to the vicinity and zoom in. Double click when you see the parking location.

Submit Trailhead Parking

Title: *

Location

Coordinates:
Lat.: °
Lng.: °
☐ You need to double click on the map to add coordinates.
☐ You can drag the marker to arrange the coordinates.



Map Satellite Hybrid

Harford Rd 147

POWERED BY Google

Imagery ©2008 DigitalGlobe, Sanborn, GeoEye, U.S. Geological Survey, Map data ©2008 Tele Atlas - Terms of Use

View setup

10. Create the View for the Map of Trailhead Parking Sites.

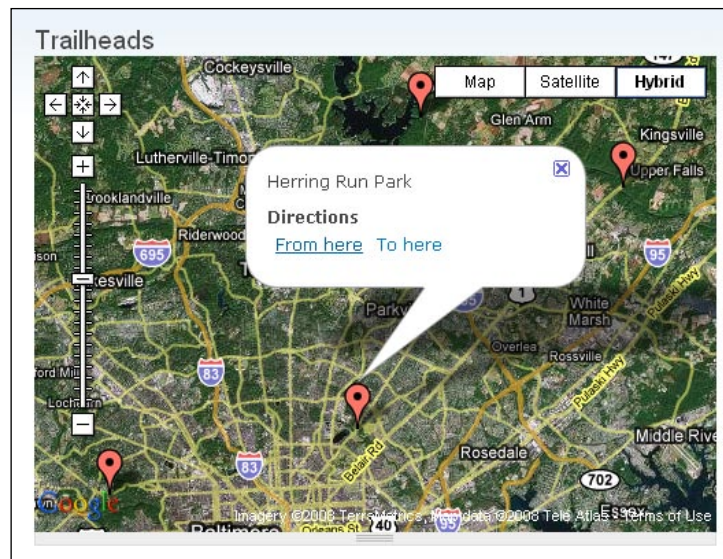
Name	trailheads
Description	Map of trailhead parking areas
Provide	Checked
Page View	
URL	trailheads

View Type	Google Maps View
Title	Trailheads
Use Pager	Checked
Nodes Per Page	50
Fields	Field list marker - Earth map - Map marker: Location (field_th_parking_location) Node Title - Option: Without link Map marker: Location (field_th_parking_location) - Directions links

Name	Label	Handler	Option
Field list marker - Earth map - Map marker: Location (field_th_parking_location)		Show on first value only ▼	
Node: Title		Normal ▼	Without link ▼
Display the title of the node.			
Map marker: Location (field_th_parking_location) - Directions links		Group multiple values ▼	

Filters	Node: Published Equals Yes Node: Type Is One Of Trailhead Parking
----------------	--

11. Visit <http://YOURSITE.com/trailheads> and enjoy the view!



Recipe notes

- There is another mapping module, also powerful, not covered here, called GMap. Take note that <http://drupal.org/project/gmap> and <http://drupal.org/project/gmaps> are two separate project pages.
- If you encounter a **packet bigger than 'max_allowed_packet' bytes** warning, it will be helpful to apply the patch at <http://drupal.org/node/218187>, or a similar patch found at <http://drupal.org/node/121390#comment-747338>. We will cover patching in more detail in Chapter 8. It may also help to increase the memory available to the Drupal site, by adding the following line:

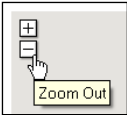
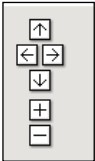
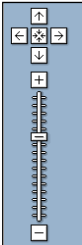
```
php_value memory_limit 64M
```

to the `.htaccess` file in the Drupal root directory.

- Create icons to match your website's colors at `admin/build/gmaps-icon-config/add/factory`.
- Control icon configurations at:
<http://YOURSITE.com/admin/build/gmaps-icon-config>.
- Configured icons may be selected per content type, per field, or per individual nodes (individual node settings override the others).

Icon configurations				
		List	Add marker icon	
Name ▲	Type	Size	Preview	Operations
Red balloon	Built-in icon	20x34		edit delete

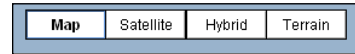
- Refer to the following visual guide to Google Maps when setting up your map configurations.

Map Controls			
Disabled	Small Zoom Control	Small Map Control	Large Map Control
			

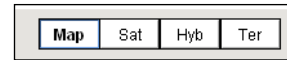
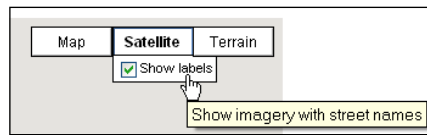
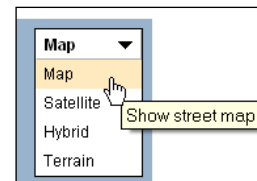
Map Type Controls

Disabled**Standard**

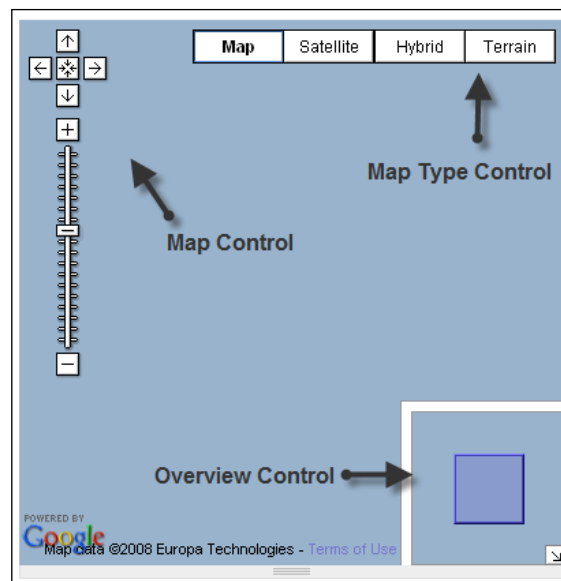
Full Text:



Short Names:

**Hierarchical****Menu**

The following screenshot displays the **Map**, **Map Type**, and **Overview** Controls together in one view:



Google Map Search

This option is found in **Google search bar options** fieldset that is closed by default in the map configuration.



Recipe 62: Proximity Search

Ingredients

Completed Recipe 61

Views Custom Field module:

http://drupal.org/project/views_customfield

In this recipe we'll create two different views. The result will be a listing of Senior Centers with a link to a view of the nearest Trailhead Parking areas.

We'll create a Senior Center content type, with a **Geocoded address marker** field (recall that the **Trailhead Parking** content type uses a **Map marker** field, the *Recipe Notes* contain a detailed comparison of the five CCK field types).

1. Complete *Recipe 61*.
2. Install and enable the **Views Custom Field** Module.

3. Create a new Map configuration at `admin/build/gmaps-map-config/add` (this is essentially the same configuration as Step 4 of *Recipe 61*, but with a different name, and the **Default Tile** changed to **Normal map**).

Map type	Earth Map
Configuration name	Street Map
Default Zoom level	6
	Note: Zoom levels range from 0 (Satellite view, or no zoom) to 19 (close-up).
Allowed Tiles	Select Normal , Satellite , and Hybrid maps Deselect Physical Map
Default Tile	Normal map
Methods	Enable dragging Enable automatic center and zoom Enable height resize
Map Control	Large Map Control
Map Type Control	Standard

4. Create a **Senior Center** content type at `admin/content/types/add`, with the following settings:

Name	Senior Center
Type	senior_center
Description	Public or privately-owned senior center
Body Field	Omit field
Promoted to front page	Uncheck

5. Add a **Geocoded Address Marker Field** to the **Senior Center** content type at `admin/content/types/senior-center/add_field`. Use the following Settings:

Name	senior_center_address
Label	Address
Point settings: Map Configuration	Street Map

The field summary at `admin/content/types/senior-center/fields` will look like this:

Label	Name	Type	Weight	Operations
title	title	title	-5	
Address	field_senior_center_address	Geocoded address marker	0	configure remove

[Update](#)

- Populate the **Senior Center** with content. Rather than filling out each individual address component field, it's easiest to do a single field address search. Upon pressing the search button, the address is standardized—even if the zip code is wrong or missing, Google Maps can probably correct it. The City (known as the **Locality** to Google Maps) may also be modified to the Postal Service's preferred address. The address is also geocoded at this time, populating the latitude and longitude fields.

Submit Senior Center

Title: *
Essex Senior Center

Address

Address and coordinates:
 600 Dorsey Avenue, baltimore, md [Search](#)
☐ Restrict search to selected country
Automatically adds the country code to the query.

Country: United States
 Admin. area: MD
 Sub-admin. area:
 Locality: Essex
 Dep. locality:
 Street, number: 600 Dorsey Ave
 Postal code: 21221

Lat.: 39.3102083
 Lng.: -76.4741305

☐ You need to double click on the map to add coordinates.
☐ You can drag the marker to arrange the coordinates.

- Create a view for Senior Centers. Here we make use of the Views Custom Field, to add a link pointing to a view which we will create in Step 9.

Name	senior_centers
Description	List of senior centers, with links to nearby trails
Provide Page View	Checked
URL	senior-centers
View Type	Table View
Title	Senior Centers and Nearby Trails
Use Pager	Checked
Nodes Per Page	30
Fields	Node Title - Option: As link (Label: Senior Center) Geocoded Address marker: Address (field_center_center_address) - Thoroughfare (Label: Street) Geocoded Address marker: Address (field_center_center_address) - Locality (Label: City) Geocoded Address marker: Address (field_center_center_address) - Admin. Area (Label: State) Geocoded Address marker: Address (field_center_center_address) - Postal code (Label: Zip) Views Custom Field: Text (Label: Nearest Trails) <?php print l("Link","trailhead-list/" . \$data->nid) ?>

Name	Label	Handler	Option
Node: Title	Senior Center	Normal	As link
Display the title of the node.			
Geocoded address marker:			
Address (field_senior_center_address)	Street	Group multiple values	
- Thoroughfare			
Geocoded address marker:			
Address (field_senior_center_address)	City	Group multiple values	
- Locality			
Geocoded address marker:			
Address (field_senior_center_address)	State	Group multiple values	
- Admin. area			
Geocoded address marker:			
Address (field_senior_center_address)	Zip	Group multiple values	
- Postal code			
Views Custom field: Text	Nearest Trails		
Value:			
<?php print l("Link","trailhead-list/" . \$data->nid) ?>			

Filters	Node: Published Equals Yes
	Node: Type Is One Of Senior Center
Sort Criteria	Node: Title

8. **Save** the view, and visit <http://YOURSITE.com/senior-centers>.

Senior Centers and Nearby Trails					
Senior Center	Street	City	State	Zip	Nearest Trails
Allen Senior Center	1404 S Charles St	Baltimore	MD	21230	Link
Bykota Senior Center	611 Central Ave	Towson	MD	21204	Link
Catonsville Senior Center	501 N Rolling Rd	Catonsville	MD	21228	Link
Cherry Hill Senior Life Center	606 Cherry Hill Rd	Baltimore	MD	21225	Link

9. We will now set up the **trailhead-list** linked view. Add a new view at `admin/build/views/add`.

Name	trailhead_list
Description	Map of trailhead parking areas
Provide Page View	Checked
URL	trailhead-list
View Type	Table View
Title	<i>(Leave blank.)</i>
Use Pager	Checked
Nodes Per Page	25
Header	<pre><h1>Trailheads within 6 miles of</pre> <pre><?php</pre> <pre>\$nid =intval(arg(1));</pre> <pre>\$node = node_load(\$nid);</pre> <pre>print \$node->title;</pre> <pre>?></pre> <pre></h1></pre> <i>Be sure to set the input format to PHP code. This Header code looks at arg(1), the URL argument that specifies the senior center node ID. It loads that ID's node, and prints the title, which is the name of the senior center. If the Argument Setting of 6 miles is modified, the first line will need to be edited accordingly.)</i>

Fields**Node Title - Option: Without link****Field list marker - Earth map - Map marker: Location****(field_th_parking_location) Option: Static Map****Map marker: Location (field_th_parking_location) - Directions to link****Provide a label each field: Park, Map, and Directions.**

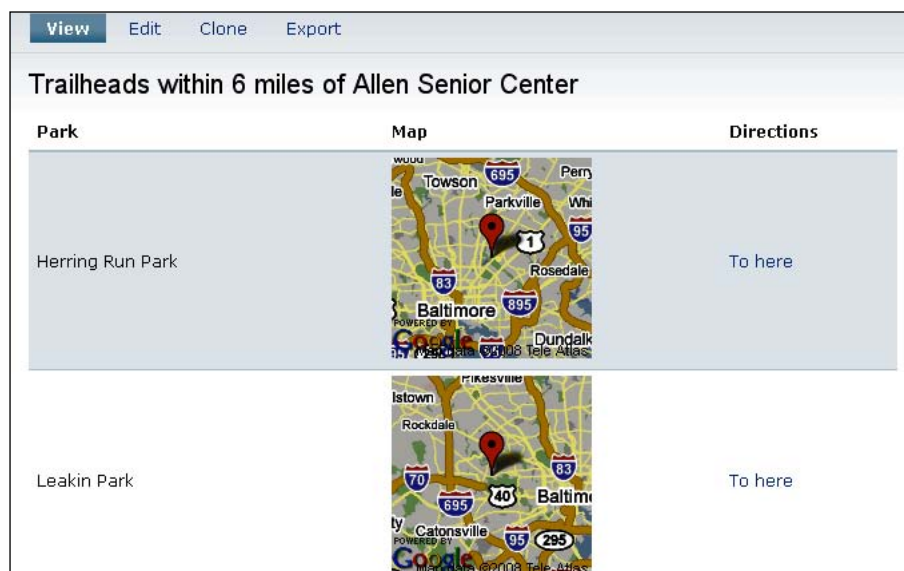
Name	Label	Handler	Option
Node: Title	Park	Normal	Without link
Display the title of the node.			
Map marker: Location (field_th_parking_location)	Map	Show first value only	Static map
Multiple, ungrouped values can be table sorted on their creation order.			
Map marker: Location (field_th_parking_location) - Directions to link	Directions	Group multiple values	

Arguments**Proximity by distance from node - Map marker: Location
(field_th_parking_location)****Max Distance: 6 Miles****Operator: Is Less Than Or Equals****Order by: Ascending****Distance: Least**

Argument Type	Default	Title	Option
Proximity by distance from node - Map marker: Location (field_th_parking_location)	Return Page Not Found		Max. distance: 6 miles Operator: Is Less Than Or Equals Order by: Ascending Distance link: Least
This argument requires a node ID and filters the content by the distance from the given node.			

Filters**Node: Published Equals Yes****Node: Type Is One Of Trailhead Parking**

10. Edit the Static Map settings at `/admin/content/types/trailhead-parking/fields/field_th_parking_location`. Set the map width and height to **150 px**.



Recipe notes

This table distinguishes the five CCK fields that come with Google Maps Tools. Note that four of the five fields, the ones that contain latitude and longitude, are mappable. The Address field just contains text variables and can not be mapped.

	Address	Geocoded Address	Geocoded Address Marker	Point	Marker
Address and coordinate Search		✓	✓		
Country	✓	✓	✓		
Admin. area	✓	✓	✓		
Sub-admin. area	✓	✓	✓		
Locality	✓	✓	✓		

	Address	Geocoded Address	Geocoded Address Marker	Point	Marker
Dep. Locality	✓	✓	✓		
Street, Number	✓	✓	✓		
Postal Code	✓	✓	✓		
Latitude		✓	✓	✓	✓
Longitude		✓	✓	✓	✓
Map		✓	✓		✓
Marker action			✓		✓
Marker URL			✓		✓
Marker title			✓		✓
Marker content			✓		✓
Input Format			✓		✓
Static map zoom		✓	✓	✓	✓
Static map marker letter		✓	✓	✓	✓
Static map marker size		✓	✓	✓	✓
Static map marker color		✓	✓	✓	✓
Map default icon		✓	✓	✓	✓
Map min zoom range		✓	✓	✓	✓
Map max zoom range		✓	✓	✓	✓

Recipe 63: Views Calculations

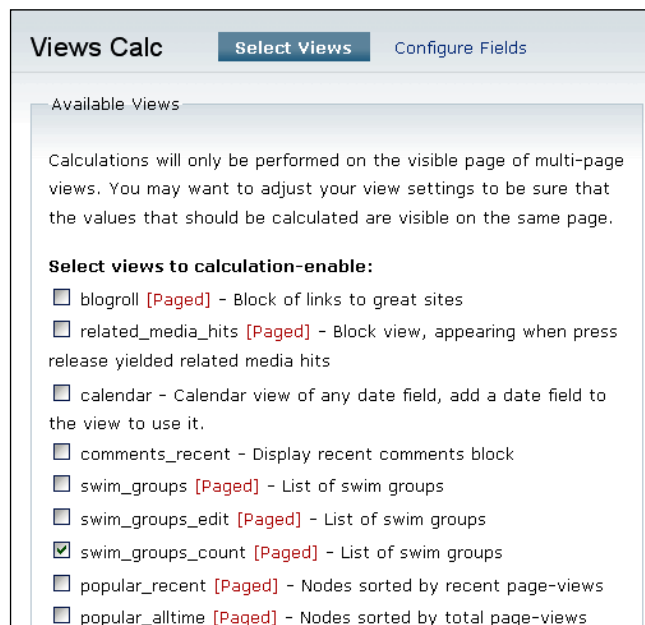
Ingredients

Completed Recipe 1, swim_groups

Views Calc module: http://drupal.org/project/views_calc

The **Views Calc** module offers sums, counts, and averages for view rows and columns. The module does its math on a page-by-page basis, so the easiest way to perform a calculation on all the items is to set the view's **Nodes Per Page** value to be equal or larger than the number of nodes in the view.

1. Install and enable the **Views Calc** module.
2. Clone the **swim_groups** view, at `admin/build/views/swim_groups/clone`. Name the view **swim_groups_count** and change the url to `swim-groups-count`. Set the view type to **Calc Table View** and ensure that at least one field is displaying in the Fields section.
3. Select the Views for which calculations will be enabled at `admin/config/views_calc`. In our case, we'll select only the **swim_groups_count** view.



The screenshot shows the 'Views Calc' configuration interface. At the top, there are two tabs: 'Select Views' (active) and 'Configure Fields'. Below the tabs, a section titled 'Available Views' contains a text box explaining that calculations are performed on the visible page of multi-page views. Below this, a section titled 'Select views to calculation-enable:' lists several views with checkboxes. The view 'swim_groups_count [Paged] - List of swim groups' is selected with a checked checkbox. Other views listed include 'blogroll [Paged]', 'related_media_hits [Paged]', 'calendar', 'comments_recent', 'swim_groups [Paged]', 'swim_groups_edit [Paged]', 'popular_recent [Paged]', and 'popular_alltime [Paged]'.

View Name	Selection	Description
blogroll [Paged]	☐	Block of links to great sites
related_media_hits [Paged]	☐	Block view, appearing when press release yielded related media hits
calendar	☐	Calendar view of any date field, add a date field to the view to use it.
comments_recent	☐	Display recent comments block
swim_groups [Paged]	☐	List of swim groups
swim_groups_edit [Paged]	☐	List of swim groups
swim_groups_count [Paged]	☒	List of swim groups
popular_recent [Paged]	☐	Nodes sorted by recent page-views
popular_alltime [Paged]	☐	Nodes sorted by total page-views

4. Configure Views Calc Fields for the **swim_groups_count** view by selecting the **Configure Fields** tab at `admin/config/views_calc/view`. Open the **View: swim_groups_count** fieldset and modify the column settings (we will leave the row settings blank). Choose the **node_title** column, for a **Count**.

Views Calc
Select Views
Configure Fields

Calculations will only be performed on the visible page of multi-page views. You may want to adjust your view settings to be sure that the values that should be calculated are visible on the same page.

Select the fields to calculate for each view that has been calculations-enabled.

View: swim_groups_count [Paged] - List of swim groups

Fields for row calculation:

node_title

Add a calculation at the end of each row, using these fields.

Calculations to perform on rows:

Sum
Count
Average
Average (all)

Columns for column calculation:

node_title

Add a calculation at the foot of each these columns.

Calculations to perform on columns:

Sum
Count
Average
Average (all)

Note: **average (all)** will divide the total by the number of all items, **average** will divide the total by the number of items that have a non-zero value.

Save configuration Reset to defaults

5. The view at [YOURSITE.com/swim-groups-count](https://yoursite.com/swim-groups-count) now contains a count of all of the swim groups.

Water Wrigglers (Ages 2 to 3)
Tadpoles (Ages 3 to 5)
Guppies (Ages 6 to 12)
Flying Fish (Ages 9 - 14)
Adult Lessons (Age 15+)
5

Recipe 64: Podcasting

Ingredients

Audio: <http://drupal.org/project/audio>
Audio GetID3 (Packaged with audio module)
Path (Part of Drupal core)
Pathauto: <http://drupal.org/project/pathauto>
Token: <http://drupal.org/project/token>
Views RSS (Packaged with Views module)
getID3 library: <http://getid3.org/>
Unzip utility

The Audio module is the centerpiece of our podcasting recipe; The module creates a custom Audio content type and comes with a helpful default view. Our view enables website visitors to play audio play via a Flash player and offers a helpful podcast feed.

Setup

1. Download and untar the audio module to the usual `<DRUPALROOT>/sites/all/modules` location.
2. Be sure to read the `INSTALL.txt` found in the new `<DRUPALROOT>/sites/all/modules/audio` directory. Step 2 of this file is important (use the web site <http://getid3.org> to help locate the getID3 library at **SourceForge.net**).

```
wget http://downloads.sourceforge.net/getid3/getid3-1.7.9.zip?use_mirror=voxel
unzip getid3-1.7.9.zip
rm -r demos/
```
3. Install and enable all of the modules listed in the ingredients.
4. It may be helpful to increase the upload size limit for your site. By default the site is usually limited to a 2 MB upload size limit. Edit the file named `.htaccess` in your site's root directory, adding the directive:

```
php_value upload_max_filesize 4M
```

Adding content

5. (Optional) Add a taxonomy category for Artist to the Audio content type, at `admin/content/taxonomy/add/vocabulary`. MP3 tags (also known as **ID3 tags**) offer a single field for Artist. If we choose to add a taxonomy field for Artist, we will actually be required to enter the artist name more than once, but it's an easy way to make sure that multiple artists can have their own page on the site. It would be hard, for instance, to offer a separate page for songs by "Naomi Bernstein" with the MP3 Artist set to "Leah Ulansey and Naomi Bernstein". Our vocabulary applies to the **Audio** Type, with settings of **Free tagging**, **Multiple select**, and **Required**.

Edit vocabulary

Vocabulary name: *

The name for this vocabulary. Example: "Topic".

Description:

Description of the vocabulary; can be used by modules.

Help text:

Instructions to present to the user when choosing a term.

Types: *

☒ Audio
☐ Link
☐ Page
☐ Slideshow
☐ Story

A list of node types you want to associate with this vocabulary.

Hierarchy:

☒ Disabled
☐ Single
☐ Multiple

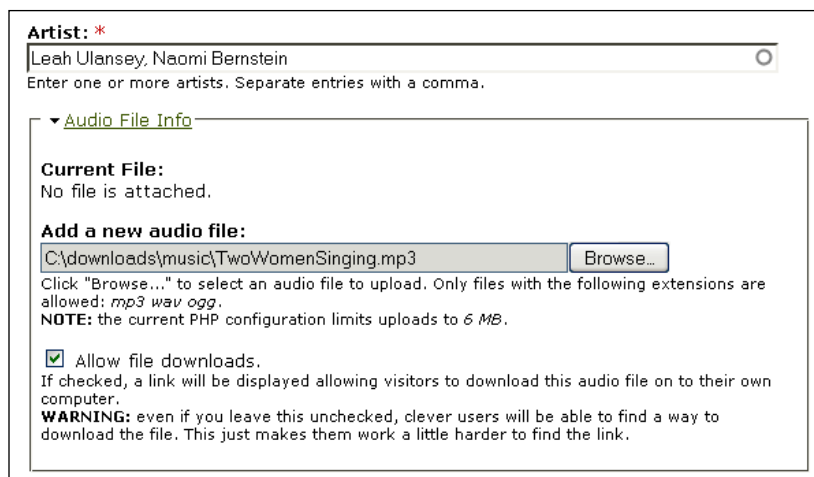
Allows [a tree-like hierarchy](#) between terms of this vocabulary.

☐ Related terms
 Allows [related terms](#) in this vocabulary.

☒ Free tagging
 Content is categorized by typing terms instead of choosing from a list.

☒ Multiple select
 Allows nodes to have more than one term from this vocabulary (always true for free tagging).

☒ Required
 If enabled, every node **must** have at least one term in this vocabulary.

6. Upload MP3 files at `node/add/audio`.

The screenshot shows a web form for adding an audio node. At the top, there is a text field labeled "Artist: *" containing the text "Leah Ulansey, Naomi Bernstein". Below this field is a small instruction: "Enter one or more artists. Separate entries with a comma." Below the artist field is a section titled "Audio File Info" with a downward-pointing triangle icon. Inside this section, there is a "Current File:" label followed by the text "No file is attached." Below that is a section titled "Add a new audio file:". This section contains a text field with the file path "C:\downloads\music\TwoWomenSinging.mp3" and a "Browse..." button. Below the text field is a note: "Click 'Browse...' to select an audio file to upload. Only files with the following extensions are allowed: mp3 wav ogg." Below this note is a "NOTE:" stating "the current PHP configuration limits uploads to 6 MB." At the bottom of the section is a checkbox labeled "Allow file downloads." which is checked. Below the checkbox is a warning: "If checked, a link will be displayed allowing visitors to download this audio file on to their own computer. WARNING: even if you leave this unchecked, clever users will be able to find a way to download the file. This just makes them work a little harder to find the link."

Upon uploading the file, the node form populates the **Audio Metadata** and **Audio File Info** fieldsets with detailed information from the MP3 file data.

Thanks to the **Token** module, the Node form automatically names the node based on the ID3-tagged artist and title. If the MP3 file does not contain this information, it can be populated via the Drupal node form (i.e. the Drupal form can actually modify the MP3 file). The screenshot that follows shows an excerpt from the node form after the file has been uploaded.



The sample rate for this particular file is 44.1 KHz. This sample rate enables the 1PixelPlayer flash player to appear. Other flash-player compatible sample rates are 22 KHz and 11 KHz. If the file is saved at a different sample rate in the audio editing software, the Flash player will not be available and will be replaced by a link to the file.

Artist: *
Leah Ulansey, Naomi Bernstein
Enter one or more artists. Separate entries with a comma.

▼ Audio Metadata

Artist: *
Leah Ulansey and Naomi Bernstein

Title: *
Two Women Singing

Album:
Baltimore Songwriters Association - From Our Circle

Track Number:
21
Enter either a single number or fraction here. '1' means that this is the first track on the album, and '1/8' .

Genre:
Folk

Year:
2001

▼ Audio File Info

This file information was loaded from the file by the getID3 library.

Current File	files/audio/TwoWomen.mp3
Play count	8
Download count	3
Format	mp3
File Size	2,877,694 bytes
Length	3:00
Sample rate	44,100 Hz
Channel mode	Stereo
Bitrate	128,000 bytes/second
Bitrate mode	CBR

Replace this with a new file:

Click "Browse..." to select an audio file to upload. Only files with the following extensions are allowed: *mp3 wav ogg*.
NOTE: the current PHP configuration limits uploads to *6 MB*.

☒ Allow file downloads.
If checked, a link will be displayed allowing visitors to download this audio file on to their own computer.
WARNING: even if you leave this unchecked, clever users will be able to find a way to download the file. This just makes them work a little harder to find the link.

View setup

- The setup for our view will be fairly easy. Go to `admin/build/views` and click on the **Add** link for the Audio default view. Review the view settings, and **Save**.
- Consider removing the post information ("Submitted by Username on date") for the audio type at `admin/build/themes/settings`.

9. Enjoy the view at <http://YOURSITE.com/audio>. You may click on an individual artist to view all of his or her pieces. Note the **Click to play** link that appears for audio files not encoded with one of the flash-enabled sample rates.

The screenshot shows a web page with a navigation bar at the top containing links: Home, Directions, Folk Music News, Audio, and Links. Below the navigation bar, there is a breadcrumb trail: Home > Audio. The main content area features three audio tracks. The first track is 'Two Women Singing by Leah Ulansey', with links to 'Leah Ulansey' and 'Naomi Bernstein'. It includes a progress bar showing 00:00:46, a duration of 3:00 minutes (2.74 MB), and statistics: 12 reads, Download audio file, 2 downloads, and 6 plays. The second track is 'La Bicicletta by A Bike Commuting Trio', with links to 'Leah Ulansey', 'Marjorie Roswell', and 'Matthew Clark'. It includes a progress bar showing 1:15 minutes (1.44 MB), a duration of 1:15 minutes (1.44 MB), and statistics: 852 reads, Download audio file, 146 downloads, and 119 plays. The third track is 'Willie Nilly sung by Margie, Leah, and Joanne', with links to 'Joanne Stato', 'Leah Ulansey', and 'Marjorie Roswell'. It includes a 'Click to play' link, a duration of 3:11 minutes (1.82 MB), and statistics: 224 reads, Download audio file, 182 downloads, and 246 plays. The lyrics for 'Willie Nilly' are displayed below the track: William D. H. Nilly, Yes, he's a friend of mine, I never know quite when I'll see him, But to see him is divine, read more >.

10. The view also enables the RSS feed for the podcast at:
<http://YOURSITE.com/audio/feed>.

The screenshot shows an RSS feed titled 'Audio'. It lists three audio nodes. The first node is 'Spanish Dance by Jonathan Rudie on mandolin, Mark Edwards on Guitar', dated Sunday, December 14, 2008 9:35 PM. It includes a media file link: SpanishDance.mp3 (MPEG Layer 3 Audio, 2.6 MB). The second node is 'Two Women Singing by Leah Ulansey and Naomi Bernstein', dated Sunday, December 14, 2008 9:01 PM. It includes a media file link: http://friendlycoffehouse.org/audio/download/27/ (MPEG Layer 3 Audio, 2.7 MB). The third node is 'La Bicicletta by A Bike Commuting Trio', dated Monday, August 04, 2008 11:26 AM. It includes a media file link: http://friendlycoffehouse.org/audio/download/24/ (MPEG Layer 3 Audio, 1.4 MB).

- Visit `feedburner.com`, and enter the feed URL. You will need to create a **FeedBurner** account to be able to track podcast statistics (visitors, downloads, and so on).



Hello, our name is FeedBurner.
Put your content in front of more eyeballs. [Publish + Advertise](#)



Blogs »
Promote content, build & measure audience, make \$

Podcasts »
Track your subscribable audio & video content

Commercial »
Join Wall Street Journal, USA TODAY, Newsweek, etc.

Blog & Feed Advertising Programs »
See why brands like BlackBerry & Verizon choose FeedBurner for blog and RSS advertising.

Start FeedBurning Now. Type your feed or blog URL below. (Why Burn?)
Learn how to burn your [Blogger](#), [TypePad](#), [WordPress](#), [MySpace](#) or [podcastVideo](#) feed.

☒ I am a podcaster!

Recipe notes

- To enable each individual artist to have his or own feed, add a **Taxonomy: Term ID** argument to the view. Be sure to make it the first argument, set the Default to **Display All Values**, the Title to **%1**, and the wildcard to **all**. Change the main FeedBurner feed to `http://YOURSITE.com/all/feed`. Individual artist feeds will have this format `http://YOURSITE.com/1/feed`.

Argument Type	Default	Title	Option	Wildcard	Wildcard Sub	Ops
Taxonomy: Term ID	Display All Values	%1		all		   
The argument will filter by a taxonomy term ID. For this argument, set the option to the depth to search. See taxonomy for more information.						
RSS: RSS Feed Selector	Display All Values					   
This argument specifies a specific RSS feed selector; it will only select RSS feeds, unlike the built-in selector which can select pluggable feeds. You may enter the title the feed will advertise in the title field here, and the description of the feed in the option field here.						

- A quick way to generate a URL alias for each artist's view is to use the View Alias module, available at http://drupal.org/project/view_alias. A sample configuration is shown below, along with the resulting aliases, which may be incorporated into a menu.

View Aliases

▼ [Generate Aliases](#)

audio

Vocabulary to alias:
 Artist ▼
 select the vocabulary that the view is intended to use as term id arguments

☒ Update aliases for **audio**.

Update action:

☐ Do nothing. Leave the old alias intact.

☒ Create a new alias. Delete the old alias.

☐ Delete aliases for selected views.

What should view alias do when updating an existing content item which already has an alias?

Recurring Aliases

audio ([audio](#))

☒ Create/Update/Delete aliases for **audio** on term creation.

Vocabulary to alias:
 Artist ▼
 select the vocabulary that the view will be aliased with

URL aliases

Drupal provides users complete control over URLs through aliasing. This feature is typically used to make URLs human-readable or easy to remember. For example, one could map the relative URL 'node/1' onto 'about'. Each system path can have multiple aliases.

[\[more help...\]](#)

Alias ▲	System	Operations
audio/debby-bors	audio/4	edit delete
audio/delete	audio/10	edit delete
audio/joanne-stato	audio/5	edit delete
audio/john-seay	audio/3	edit delete
audio/jonathan-rudie	audio/7	edit delete
audio/leah-ulansey	audio/2	edit delete
audio/marjorie-roswell	audio/1	edit delete
audio/mark-edwards	audio/9	edit delete
audio/matthew-clark	audio/6	edit delete

Recipe 65: Using the Flag module for Bookmarks

Ingredients

Flag module: <http://drupal.org/project/flag>

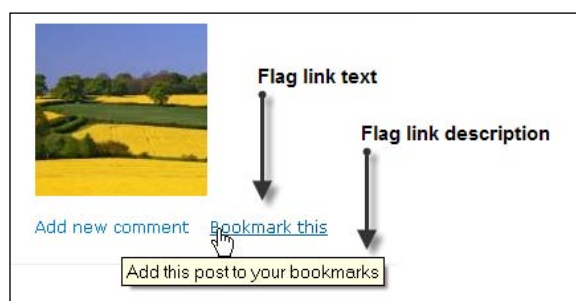
Optional: Completed Recipe 59

In essence, the **Flag** module links a user to a node, allowing users to mark content for later viewing. Unlike the audio module, which is basically designed to do one thing (podcasting), the **Flag** module may be applied to many different use cases. In this recipe we apply the default **flags_bookmarks** view, serving end users. In *Recipe 66*, we add fields, arguments, and filters, to customize the defaults for various applications serving site administrators.

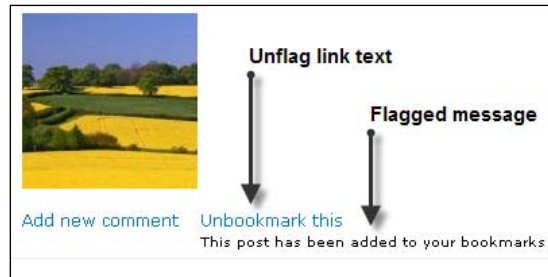
Default Bookmarks View

1. Install and enable the **Flag** module.
2. Configuration for this module resides in the **Site building** area (we're typically used to seeing configuration under **Site configuration**). Visit `/admin/build/flags`, and select **edit** to configure the default **bookmarks** flag. Review the settings to become familiar with the options. For now we will only make a single modification: select the **Photo** content type (or the content type of your choice) to the list of **nodes this flag may be used on**.
3. Visit a sample node (of the content type that you selected in Step 2). You will see a new link: **Bookmark this**.

Upon mousing over the link, a pop up Flag link description appears.



When the link is clicked, it toggles to display the Unflag link text, **Unbookmark this**, along with a flagged message.



4. Visit and bookmark several more photos (you may find the photos by filtering by content type at `admin/content/node`, or via the **photos** view created in *Recipe 59*).
5. Add the **flag_bookmarks** view (a default view) at `admin/build/views`. Review the view settings and click on **Save**.
6. Visit `http://YOURSITE.com/bookmarks` and enjoy the view, which has been customized for each individual user.

My bookmarks					
View Edit Clone Export					
Type	Title	Author	Replies	Last Post▼	Ops
Photo	Canola Field	admin	0	2 weeks 1 day ago	Unbookmark this
Photo	Maze	admin	0	2 weeks 1 day ago	Unbookmark this
Photo	Autumn Colors	admin	0	2 weeks 1 day ago	Unbookmark this

Recipe 66: Using the Flag module for various use cases

Ingredients

Flag module: <http://drupal.org/project/flag>

Views Custom Field module: http://drupal.org/project/views_customfield

In this recipe, we expand upon *Recipe 65*, examining more use cases for the Flag module with Views. We'll manage fact-checking by editors and prepare lists of workshop attendee RSVPs. We'll also flag some comments as inappropriate and see one of the limits to the views module in that context.

1. Complete *Recipe 65*. Additionally, install and enable the **Views Custom Field** module.
2. Create three flags: **inappropriate**, **fact_check**, and **rsvp**), as follows, at `admin/build/flags/add`. The **bookmarks** flag settings that come with the module are shown for comparison.

Note that the table of settings continues on the following page:

Name	bookmarks	inappropriate	fact_check	rsvp
Type	Nodes	Comment	Nodes	Nodes
Title	Bookmarks	Inappropriate	Fact Check Needed	RSVP
Flag link text	Bookmark this	Flag as Inappropriate	Fact Check	RSVP for this Event
Flag link description	Add this post to your bookmarks	Flag this as an inappropriate comment	Reminder to do further research on this content	
Flag confirmation message				Are you sure you can attend?
Flagged message	This post has been added to your bookmarks	This has been flagged as an inappropriate comment	Flagged for fact-checking	You have RSVPd for this event
Unflag link text	Unbookmark this	Unflag this as Inappropriate	Unflag for fact-checking	Regrets – I can't attend afterall.
Unflag link description	Remove this post from your bookmarks		Mark Fact Check Complete	Please let us know if you can not attend
Unflag confirmation message				Are you sure you are unable to attend?
Unflagged message	This post has been removed from your bookmarks	This post is no longer flagged as inappropriate	Fact-checking is complete	We'll miss you. Hope you can attend a future workshop!

Global flag	Not checked	Checked	Checked	Not checked
Roles that may use this flag	authenticated user	admin	editor	authenticated user
What nodes this flag may be used on	photo, story (or content type of your choice)	photo, story (or content type of your choice)	timeline_item (or content type of your choice)	Workshop (or content type of your choice)
Display link on node teaser	Not checked	—	Checked	Checked
Display link on node page	Checked	—	Checked	Checked
Display checkbox on node edit form	Checked	—	Checked	Checked
Display Link under comment	—	Checked	—	—
Link Type	Javascript toggle	Javascript toggle	Javascript Toggle	Confirmation form

The **Flags** page will look something like this (your selection of node types may vary to suit your tastes):

Flags

List

Add

This page lists all the *flags* that are currently defined on this system. You may [add](#) new flags.

Flag	Flag type	Roles	Node types	Global?	Operations	
bookmarks	node	authenticated user	story, photo	No	edit	delete
inappropriate	comment	admin, editor	photo, story	Yes	edit	delete
fact_check	node	editor	timeline_item	No	edit	delete
rsvp	node	authenticated user	workshop	No	edit	delete

Viewing a List of inappropriate comments

3. The comment flag type is a bit of a dead-end in Drupal 5, in terms of the Views 1 module. Given Views 1 module constraints, we cannot generate a view of comments flagged as inappropriate. Thankfully, on a page describing the limits of the Flag module in Drupal 5, a user has submitted SQL code that offers an alternative. The solution at <http://drupal.org/node/303589> may be copied into a PHP block, at `admin/build/blocks`. Consider limiting the block to only appear in the footer of `admin/content/comment`.

Creating a view for Fact-Checkers

4. If you do not already have an **editor** role, create one at `admin/user/roles`. Create a user account for a new user at `admin/user/user/create`, and assign it the **editor** role.
5. Assign permissions for the editor at `admin/user/access`. Provide all permissions for **Content**, **Flag**, **Node**, **Views**, and **Views UI** modules, along with the **admin_menu** permissions, if you have the **Administration menu** installed.
6. Log into the site as a user who has the editor role.
7. Flag some of your content for fact-checking by visiting several nodes (or creating new nodes) and clicking the **Fact Check** link.
8. Visit `admin/build/views` and look for the `flag_bookmarks` view that we created in *Recipe 65*. Click the **Clone** button.
9. The **flag_bookmarks** view serves as a good starting point, but it requires a number of changes, as follows.

Name	flag_fact_check
Access	Uncheck Authenticated User Check Editor and Admin
Description	Timeline items that require fact-checking (Substitute the content type name that you selected in step 2.)
URL	fact-check
Title	Delete My bookmarks Replace with Items to be Fact-Checked . (Note that the fact_check and upcoming rsvp views allow editors or administrators to see each others' flagged content. Accordingly, we will modify the flag filter from only the Currently Logged In User to Any User who has flagged content. When the Any User filter is selected, it's appropriate to remove the word <i>My</i> from the Title.)

- | | |
|----------------|---|
| Fields | <p>Delete the following three fields:</p> <p>Comment: Count (If the comment module is enabled)</p> <p>Comment: Last Comment Time (If the comment module is enabled)</p> <p>Flag: Ops for Bookmarks</p> <p>Add Flag: Ops for Fact-Check Needed</p> |
| Filters | <p>Node: Published (No changes here)</p> <p>Note that Flag: Bookmarks is set to be flagged by the Currently Logged In User. Delete this filter.</p> <p>Add Flag: Fact-Check Needed. Set the Value to be flagged by Any User.</p> |

Field	Operator	Value	Option	Ops
Node: Published	Equals ▾	Yes ▾	Expose	⌵ ⌴ ⬇ ⬆ ⬇
Filter by whether or not the node is published. This is recommended for most Views!				
Flag: Items to be Fact-Checked	Is Flagged By ▾	Any User ▾	Expose	⌵ ⌴ ⬆ ⬆ ⬇ ⬆
This will filter a only nodes that have the <i>Items to be Fact-Checked</i> flag. If using <i>Any User</i> , often the <i>Node: Distinct</i> filter will also be desired. The <i>Any User</i> filter is not necessary if combining with other flag sorts or arguments.				

10. Save the view, and begin fact-checking items listed at http://YOURSITE.com/fact_check.
11. Edit the view, and try different settings for the **Flag: Ops for Fact-Check Needed** field options. You may control whether a confirmation button click is required, and whether the flagged item disappears from the page as soon as you unflag it. After testing the options, and choosing a favorite, you may want to tweak the text settings at `admin/build/flags/edit/fact_check`.

View of RSVPs

12. Log in to the site as an administrator.
13. Flag some events or workshops as RSVPs (check or modify your flag configuration for content types that are available for flagging).
14. Visit `admin/build/views`, and look for the **flag_bookmarks** view that we created in *Recipe 65*. Click the **Clone** button.

15. The **flag_bookmarks** view serves as a good starting point, but requires a number of changes as follows:

Name	flag_rsvp
Access	Uncheck Authenticated User Check Admin
Description	RSVPs for all workshops, or individual workshops
URL	rsvp
Title	Delete My Bookmarks Add RSVPs .
Fields	Delete the following four fields: Node: Author Name Comment: Count (If the comment module is enabled) Comment: Last Comment Time (If the comment module is enabled) Flag: Ops for Bookmarks <i>(It's always a good idea to click the Save and edit button after removing a set of fields.)</i> Add the Flag: User for RSVP field. Set the Label to Attendee. Add a custom field, with the following code: <pre><?php if (!arg(1)) { print l("View Attendance", "rsvp/" . \$data->nid); } if (arg(1)) { print l("View All Workshops", "rsvp"); } ?></pre> Leave the Label blank for this field.
Arguments	Add the Node: ID argument Set the Default to Display All Values Set the Title to RSVPs for %1 Leave the Option as equal.

Argument Type	Default	Title	Option	Wildcard	Wildcard Sub	Ops
Node: ID	Display All Values	RSVPs for %	equal			

This argument is a single Node ID.

Filters	Node: Published (No changes here.) Delete the Flag: Bookmarks filter. Add Flag: RSVP . Change the Value for Flag: RSVP from Is Flagged by Currently Logged In User to Is Flagged by Any User . (Optional) Add a filter for Datestamp: workshop_date - date Set the Date Operator to greater than or equal to Set the Option to now .
Sort	Sort by the Node: Title field, or by the workshop date.

16. **Save** the view, and check your attendee lists for all workshops at <http://YOURSITE.com/rsvp>. Click on the **View Attendance** link to view attendees for a particular workshop. Click on the **View All Workshops** link to return to the attendee list for all workshops.

Recipe notes

- The **flag_rsvp** view displays all upcoming workshops by default, along with the user name of each attendee. The **View Attendance** link displays if the URL is <http://YOURSITE.com/rsvp> (i.e. if there is no argument). If a **Node: ID** argument is present, such as <http://YOURSITE.com/rsvp/205>, attendance is displayed for the single workshop and the link changes to **View All Workshops**.
- The **Node: ID** argument used in the **flags_rsvp** view is defined by the `views_argument` hook in the `views_node.inc` file. The `views_node.inc` file is usually located in `<DRUPALROOT>/sites/all/modules/views/modules`. This file is well-worth printing out and studying by those who wish to develop a more advanced understanding of views.
- The `views_node.inc` file also implements the `views_table` hook. You will recognize the creation of many basic fields, filters, and sorts in this file, such as **Node: Title**, **Node: Type**, **Node: Published**, and **Random**.
- A less commonly used node filter made available in the `views_table` hook may come in handy for some flag applications. Selecting the **Node: Distinct** filter calls the `views_handler_filter_distinct` function, thus adding a `GROUP BY node.id` to the view query. The result is that even if more than one user flagged the content, each node will appear only once in the view. (**Node: Distinct** won't be necessary for global flags, or views that use the **Flagged by Currently Logged In User** filter value. Those cases are already limited to a single node.)
- See *Appendix D* for a list of Views 1 hooks.

Recipe 67: Explore more Views, Galore

This book unfortunately can not cover all that's possible with the full array of views-enabled modules. This recipe has only one step: **Explore!** Here are a few examples of exploration routes, with screenshots to fuel your explorer's instinct.

Views Checkboxes

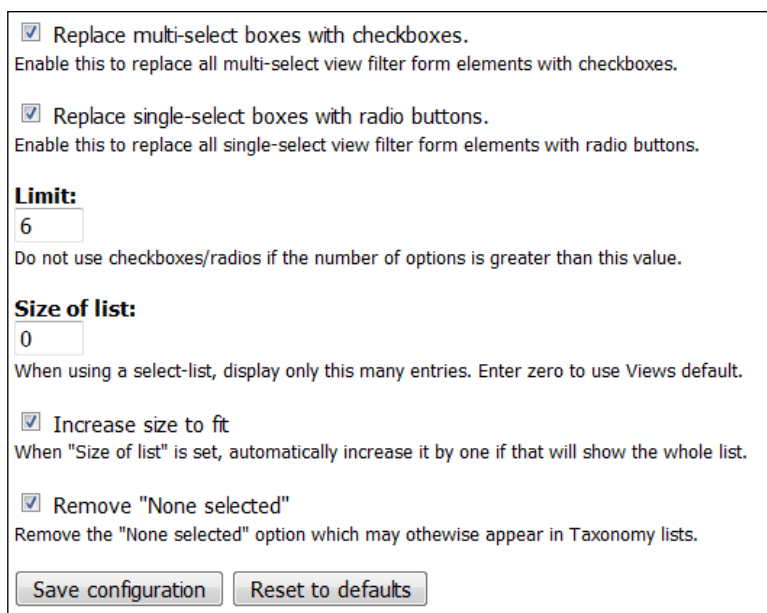
Replace select lists in Views Exposed Filters with checkboxes or radio buttons with the **Views Checkboxes** module, available at: http://drupal.org/project/views_checkboxes.



Select Activities

- ☐ Arts and Crafts
- ☐ Croquet
- ☐ Pool
- ☒ Swimming
- ☐ Table tennis
- ☐ Volleyball

Configuration for this exposed filter is shown below:



☒ Replace multi-select boxes with checkboxes.
Enable this to replace all multi-select view filter form elements with checkboxes.

☒ Replace single-select boxes with radio buttons.
Enable this to replace all single-select view filter form elements with radio buttons.

Limit:
6
Do not use checkboxes/radios if the number of options is greater than this value.

Size of list:
0
When using a select-list, display only this many entries. Enter zero to use Views default.

☒ Increase size to fit
When "Size of list" is set, automatically increase it by one if that will show the whole list.

☒ Remove "None selected"
Remove the "None selected" option which may otherwise appear in Taxonomy lists.

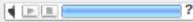
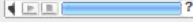
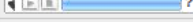
Phoneblogz

The little-known **PhoneBlogz** module, <http://drupal.org/project/phoneblogz>, allows site users to dial up by phone, and leave verbal feedback. **MarkFiore.com**, for instance, offers a "shout back!" section for users who may call to share their thoughts on his awesome political animations.

Ever wanted to yell back at these animations,
or just let the world hear your voice?
Now you can!

Just call toll-free:
1-877-415-TOON
and say what's on your mind.

Chances are, we'll publish it right here!
Rant, rave, vent-- just try and keep it clean for the kiddies.
(Don't forget to hit the # key on your phone after
you leave your message of two minutes or less.)

Title	Date/Time	Shoutback Message
TaxpayersAgainstBigOil.com!	2008-12-08 15:00	A new audio message has been left. CLICK HERE to download it or listen here: 
GM Worker	2008-11-21 18:30	A new audio message has been left. CLICK HERE to download it or listen here: 
Happy Obamaphile	2008-11-05 04:45	A new audio message has been left. CLICK HERE to download it or listen here: 
Ballot Measures	2008-11-03 23:30	A new audio message has been left. CLICK HERE to download it or listen here: 
Pretty Cool	2008-10-23 00:40	A new audio message has been left. CLICK HERE to download it or listen here: 

Default Views

In your explorations, be sure to mine the vast collection of views that come with contributed modules. The screenshot below includes a listing of all the default views that comes with the **Organic Groups** module, for instance. In Chapter 2, we covered default views that come packaged with the Views module. See *Appendix A* for a list of more default views and their associated modules.

Default Views						
Default view	Title	Description	Provides	URL	Status	Actions
og	Groups	groups directory	Page, Menu	og/all	Enabled	Add Disable
og_comments_recent		Display recent comments block in a group	Block	No Page View	Enabled	Add Disable
og_ghp_ron		OG: Group home page - River of news. Default	Page		Enabled	Add Disable
og_my	My groups	List user's subscribed groups and provide link to manage subscription	Page, Menu	og/my	Enabled	Add Disable
og_mytracker	My recent	Shows all activity in subscribed groups.	Page, Menu	group/mytracker	Enabled	Add Disable
og_recent_type_term		Recent posts for current term filtered by node type and/or taxonomy term	Page		Enabled	Add Disable
og_tracker	Recent posts	Shows active posts across the whole site	Page, Menu	group/tracker	Enabled	Add Disable
og_unread	My unread	Shows unread posts in subscribed groups.	Page, Menu	group/myunread	Enabled	Add Disable

Summary

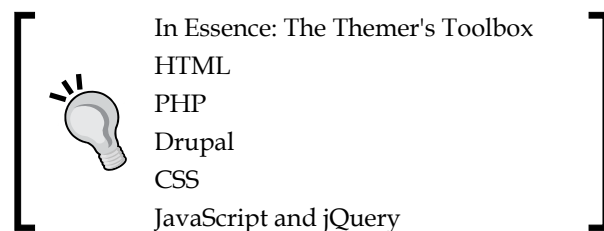
The Views Galore chapter offers a smorgasbord of Views, including fused views, enhanced searching, photo galleries, maps and proximity searches (sometimes called "Find the Nearest"), calculations, and podcasting (Ah, music!). With the Flag module, we can be quite creative about the manner in which users mark content for viewing. Our last recipe includes only one step: "Explore", with some examples of worthy approaches.

7

Theming and Layout

There's no doubt about it, this is the "deepest" section of the book. We'll see at least four types of code in this chapter, and they all interact. We have got HTML (such as `<ol>` for an ordered list). We have got PHP (such as `print '<ol>';`). We have got Drupal-specific code (`theme('views_more', $view->real_url);`) and we have got **Cascading Style Sheets** or **CSS** (`border: 1px solid #3f5f99; /*Blue */`). Indeed, there is a lot to know about if you wish to fine-tune the appearance of your views.

Additionally, some of our recipes make use of jQuery (**Views Carousel**, and **Lightbox2** both rely in the **jQuery update** module), but we won't be covering jQuery in any depth.



If code makes you pale, check out the two code-free recipes in this chapter: Panels, *Recipe 70* (also not for the "faint of heart" but there's no code) and Quicktabs Dashboard, *Recipe 85*.

But even if you've never looked at code in your life, it can't hurt too much to wade in and try all the recipes in the chapter. You may experience an *Aha!* moment.

We'll start by creating a directory of themeable functions.

Recipe 68: Themeable functions reference

[Ingredients]

No additional module are required

One of the great powers of Drupal is "themeable functions" – default functions which output HTML, which can be overridden by the theme. The following API page contains a list of selected themeable functions that come with Drupal core.

<http://api.drupal.org/api/group/themeable/5>

This recipe generates a much more comprehensive list of themeable functions from your site, including functions from contributed modules, and a link to an API page. The link will not be valid for all functions (since the contributed modules API site does not include all Drupal files), but it's an overridingly helpful block of code.

- Place the following code into a block or a story. Be sure to remember to change the **Input format** to **PHP Code**.

```
<?php
  print '<ol>';
  $functions = get_defined_functions();
  foreach($functions['user'] as $function) {
    if(substr($function, 0,6)=='theme_')
      print "<li><a href='http://api.freestylesystems.co.uk/api/
        function/$function/5' target='_blank'>$function</li>";
  }
  print '</ol>';
?>
```

- If it's a block, place the block in a region, at admin/build/blocks, and visit a page containing the block. If the code is a content item, view the content. The number of themeable functions will vary according to your site.

158. theme_views_view_table
159. theme_views_view_teasers
160. theme_views_view_nodes
161. theme_views_view_
162. theme_views_more
163. theme_views_summary
164. theme_views_nodate
165. theme_views
166. theme_devel_variable
167. theme_content_view_multiple_field

- Click on some of the links, and explore the theme functions.

Recipe notes

- The code above selects all the defined functions that start with **theme_**, and then prints them with a link to the Contributed Module API site. Each API page displays the function signatures and code, along with parameter definitions, and a link to other places where the function is called.
- The Devel module also includes a similar function reference (not just theme functions) at <http://YOURSITE.com/devel/reference>. The links, however, lead to the Drupal API page. Since this does not contain contributed modules, any contributed modules functions will lead to **Page not found**.
- The Drupal handbook uses the spelling **themeable** with an "e". Many people prefer **themable**. When doing web searches on this topic, try both spellings. In fact, you may try even five variations! **Theming function**, **themeing function**, **theme function**, and **themable function** are all sometimes used as a synonym for **themeable function**.
- Just to add to the fun: there is an actual function named `theme()`. We'll see how the `theme()` function is used to call themeable functions, in *Recipe 76*.

Recipe 69: Displaying Themeable function names in Source Code

Recipe 68 provided a reference to all of the site's themeable functions. Yet even with a comprehensive list of available functions, it can sometimes be a little tricky to determine exactly where in the site the functions are used. This recipe will make these functions easier to track down. This temporary change to core Drupal code should only be used on a development site, not on a live site.

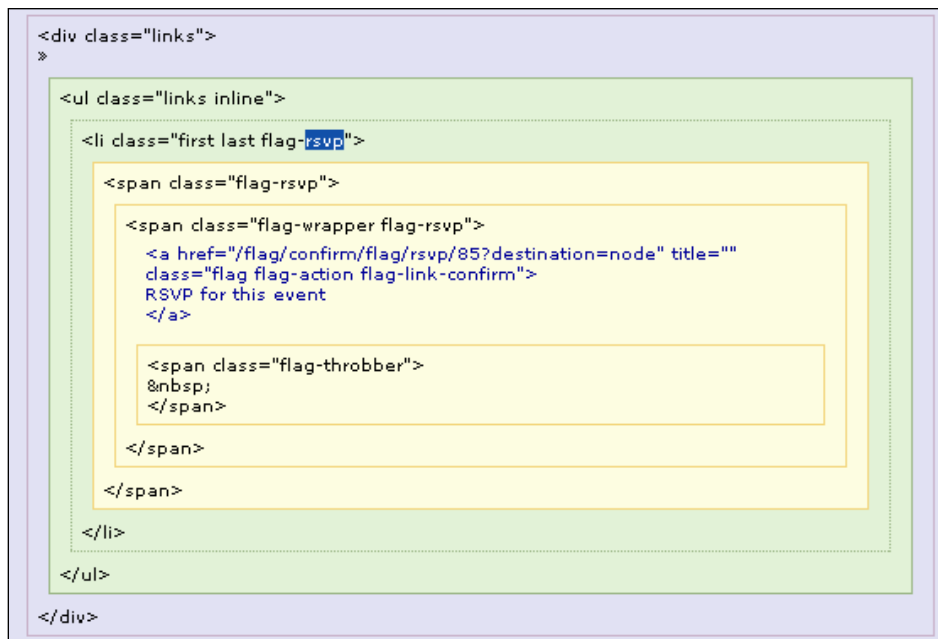
1. Make a backup of `theme.inc` located in the Drupal `includes` directory.
2. Edit the **theme** function found on line 162 of `theme.inc`. Replace this line:
`$output = call_user_func_array($functions[$function], $args);`
 with the following two lines:
`$output = call_user_func_array($functions[$function], $args);`
`return "<!-- BEGIN theme_{$function} -->\n$output\n<!--END theme_{$function} -->\n";`

3. View the source code of a page a browser (in Firefox or Chrome, press *Ctrl+U*, or right-click the page, and choose **View Page Source**). Then do a find (*Ctrl+F* in most browsers) for the word "BEGIN". Or, if you know what function you're looking for, search for that.
 - This code, while it works beautifully, may also issue a warning message. Make a copy of the code, so you can use it when needed, and always restore the original file—especially before making updates to the live site.

Recipe notes

While talking about source code for themers, three terrific Firefox add-ons are worthy of mention.

- Firebug (<http://getfirebug.com/>) is widely known, and essential.
- The Web Developer Toolbar (<https://addons.mozilla.org/en-US/firefox/addon/60>) offers a set of features with some overlap of Firebug, but most themers will be happy to have both.
- The View Source Chart add-on (<http://jennifermadden.com/scripts/ViewRenderedSource.html>) offers friendly visualization of nested `<div>` blocks, and other HTML markup, including the associated IDs and classes.



Recipe 70: Using Panels with SimpleFeed

Ingredients

Panels: <http://drupal.org/project/panels>

Enable the Panels, Panel pages, and View panes modules

SimpleFeed: <http://drupal.org/project/simplefeed>

Enable the SimpleFeed and SimpleFeed Item modules

SimplePie: <http://simplepie.org>

Unzip utility

In this recipe, we feed bicycle sale listings from **craigslist** into Drupal content items, and then use the Views and Panels modules to lay them out on the page. If you implement this recipe, please become familiar with the craigslist *Terms of Use* <http://www.craigslist.org/about/terms.of.use>.

Setup

1. Install and enable the **Panels** and **SimpleFeed** modules. Check the Ingredients for the five modules that need to be enabled for this recipe. The **SimpleFeed** module creates two content types: **feed** and **feed_item**.
2. Download the **SimplePie** program from SimplePie.org, and extract the `simplepie.inc` file into the `<DRUPALROOT>/sites/all/modules/simplefeed` directory.
3. We don't want our site to allow comments on craigslist posts. Set the **Default comment setting** to **Disabled** at both `admin/content/types/feed` and `admin/content/types/feed-item`.

Default comment setting:

- ☒ Disabled
☐ Read only
☐ Read/Write

4. There are more than a dozen permissions among the various **Panels** and **SimpleFeed** modules that we have installed. Become familiar with them at `admin/user/access`.

Creating Feed Content

- Go to <http://baltimore.craigslist.org/> and click on the **bikes** link in the **for sale** section.
- Search for **tandem**.



baltimore craigslist > bicycles [help] [post]

search for: tandem in: bicycles Search ☐ only search titles

price: min max ☐ has image

Found: 3 Displaying: 1 - 3

Jan 31 - [Two Person Bike Tandem Bicycle - \\$60](#) - (harford Co. Joppa, MD) [pic](#)

Jan 30 - [TAG A LONG FOR TWO KIDS!!! to put on YOUR bike - \\$495](#) - (HUNT VALLEY, TIMONITUM)

Jan 11 - [Trailmate Easy Ride Tandem Bicycle for two - \\$120](#) - (Severn) [pic](#) [img](#)

Found: 3 Displaying: 1 - 3

Copyright © 2009 craigslist, inc.
[Back to top of page](#) [RSS \(?\)](#) [add to My Yahoo!](#)

- Click on the **RSS** feed icon at the bottom of the page, and copy the URL from the browser into the clipboard. In this case, the URL is <http://baltimore.craigslist.org/search/bik?query=tandem&minAsk=min&maxAsk=max&format=rss>.
- Go back to the Drupal site and create a feed at `node/add/feed`.



It's tempting to try to create a feed at `admin/settings/simplefeed`, but there's no feed "add" link on the settings page (we'll leave all the settings as the default, for now). It's also tempting to try to create a feed at `admin/content/feed`, but there's no "add" button there either. A feed is a special content type.

Enter a name, a description, and paste the **RSS** URL.

Submit Feed

Title: *
Craigslist Feed - Tandem Bikes

Description: *
RSS Feed

— ▷ Input format —

URL: *
http://baltimore.craigslist.org/search/bik?query=tandem&minAsk=min&maxAsk=max&format=rss
The URL for this feed.

9. Once the feed has been submitted, the RSS feed node will include new links at the top: **Refresh Feed** and **Empty Feed** (if the **SimpleFeed Statistics** module is enabled, there will also be a **Feed log** link).

Click on **Refresh Feed**.

Now we've got new content! The content is found in another content type, called **Feed Item**.

10. Repeat Steps 4 to 8 once or twice more. Try, for instance, search terms such as **Recumbent**, or **Folding | Folder** (that last search term will find **Folding** or **Folder** bikes).

Create Views

We're going to take a simple approach and create two separate views, one for Tandems, and one for Folding Bikes.

11. First, we'll make note of the node IDs of the feeds. Visit the **Content** page at `admin/content/node` and click the **type** radio button. Select **Feed**, and press the **Filter** button. Move the cursor over the Tandem and Folding Bike feeds and make note of the Node ID in the browser's status bar (usually in the lower left). It is helpful to actually write down the ID numbers and their feed names.
12. Create two separate views. The easiest way to do this is to create the view first, clone it, and then make the necessary changes in the second view (don't forget to change the **Parent Feed Node ID**, along with the other settings, in the cloned view). For the most part, this is a basic view.

The item requiring special attention is the **Parent Feed Node ID**. Use the Node IDs determined in the previous step.

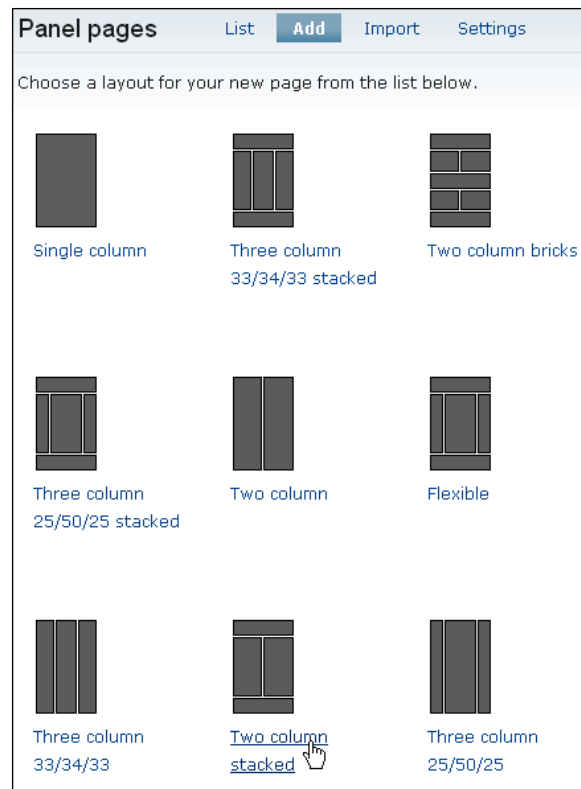
Name	tandem	folding
Description	Tandem bikes from craigslist	Folding bikes from craigslist
Provide Page View	Checked	Checked
URL	tandem	folding
View Type	Teaser List	Teaser List
Title	Tandem Bikes	Folding Bikes
Use Pager	Checked	Checked
Nodes Per Page	5	5
Filters	Node: Published Equals Yes Node: Type Is One Of Feed Item SimpleFeed Feed Item: Parent Feed Node ID is All of (Select the correct Parent Feed Node ID)	Node: Published Equals Yes Node: Type Is One Of Feed Item SimpleFeed Feed Item: Parent Feed Node ID is All of (Select the correct Parent Feed Node ID)
Sort Criteria	Node: Created Time Descending	Node: Created Time Descending

Enabling specific Views for Panels

13. In Panels Version 1, all views are available to panels by default. In Panels 2, the Views Pane module must be configured to enable specific views. Go to `admin/panels/views` and add both the **tandem** and **folding** views panes (for both views panes, click the **Create panel view** button, review the settings, and click on **Save**).

Panels setup

14. Go to `admin/panels/panel-page/add`. Review the available options, and choose **Two column stacked**.

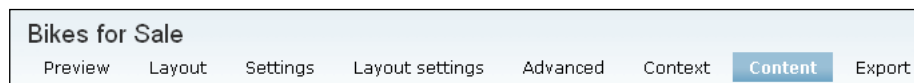


15. This takes you to the **Add Panel** page. Enter the name, title, CSS ID, and Path.

Panel Name	bikes_for_sale
Page Title	Bikes for Sale
CSS ID	bikes-for-sale
Path	bikes-for-sale

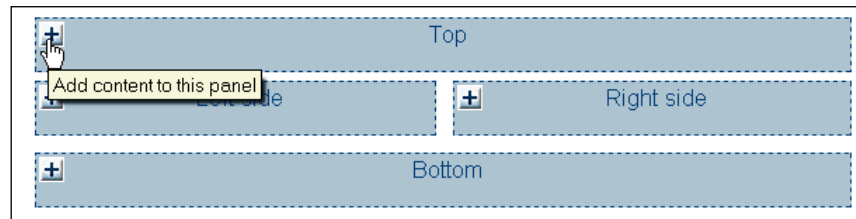
Note that all four panes (**Top**, **Left side**, **Right side**, and **Bottom**) are described on the right portion of the screen as **Empty**.

16. Click on the **Save and proceed** button.
17. The Panels interface includes eight tabs.

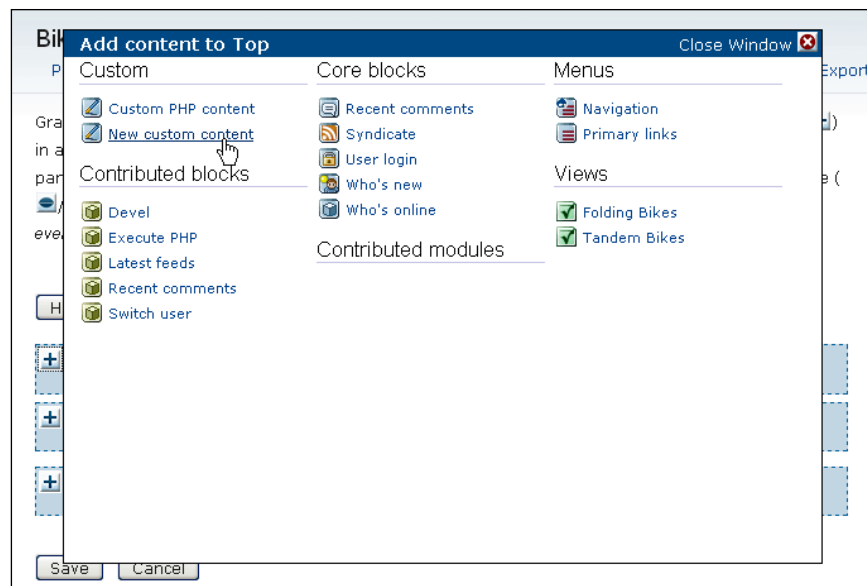


The **Preview** will be empty until we fill the panels with content. **Layout** and **Settings** contain the options that we have already set.

Let's start in the **Content** tab where we'll enter content for the **Top**, **Left side**, and **Right side** (we'll leave the **Bottom** pane empty). Click on the Plus (+) sign in the **Top** pane.



18. The **Add Content to Top** popup appears. Explore the available options and choose **New Custom Content**.



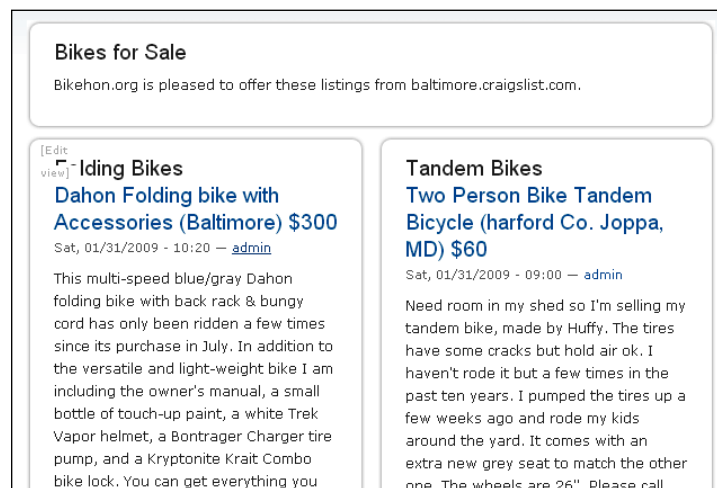
19. Enter a **Title** and a **Body** and click on the **Add pane** button.

Title: Bikes for Sale

Body: Bikehon.org is pleased to bring you sale listings from baltimore.craigslist.com.

20. In the left pane, click on the plus button, and, add the **Folding Bikes** view (only the views specified in Step 12 are available to this list).

21. In the **Right** pane, click on the plus button, and add the **Tandem Bikes** view.
22. Click on the **Save** button.
23. Click on the **Preview** tab and note that we now have two **Bikes for Sale** headings.
24. To rectify this, go to **Layout Settings** tab and check **Hide title**. While there, change the **Default Panel Style** to **Rounded corners**. Press the **Save** button.
25. Go back to the **Preview** tab, and click on the link to the **Real panel URL**, bikes-for-sale (note that the panels page provides helpful mouseover links for editing views).



26. If you wish to limit how many items display in each pane, return to the views panes settings at `admin/panels/views`. **Edit** each pane and set the **Items to display** to a small number, such as 3 or 4 (you may even place just one node into a pane).



Each views pane defaults at first to the number of nodes that we set up in the source View definition. However, the **Items to display** setting determines the number of nodes that display in the pane. Modifying the View's **Nodes per page** after the pane was created will not modify the **Items to display** pane setting after it's been created.

27. This is a good time to read the other pane options, as well, before clicking on **Save**.

Placing Panels on the Front Page

28. We could add our panels URL to a standard menu. In this case, we will move our panels to the site's home page. Go to `admin/settings/site-information` and set the **Default front page** to `bikes-for-sale`.

Recipe notes

- Given the variety of feed formats, expect to spend some time in the issue queue to successfully import your feeds. Some issues are not feed-specific. For instance, several people have reported issues with SimpleFeed and Poormanscron.
- A patch at <http://drupal.org/node/372498> prevents the need to remember the **Parent Feed Node ID**. If this gets incorporated into the module, Step 11 will not be required.
- An alternative to **SimpleFeed** is **FeedAPI**. The two modules do not work together. You must entirely uninstall one module set to get the other to work. Disable the modules and then uninstall them at `admin/build/modules/uninstall`.
- There's a nice comparison of six different feed modules at <http://groups.drupal.org/node/4547>. The content is somewhat dated, but the page is a wiki, so you may update it with your wisdom.
- Brief Panels 2 documentation comes with the module at `admin/panels`. Additional helpful screencasts are available at <http://blip.tv/file/600413/> and <http://blip.tv/file/735666/> and <http://blip.tv/file/1004793/>.
- The best way to understand panels is to experiment. Try dragging panes to different locations. Try various offset settings in separate Views panes. Using offsets, a single view could be the source for Top, Left, Middle, and Right panes.
- Coders will want to look at the two Views include files that come with the simplefeed module: `simplefeed_item_views.inc` and `simplefeed_views.inc`. Here you will find implementations of `hook_views_tables`, `hook_views_arguments`, and good old `hook_views_default_views`.

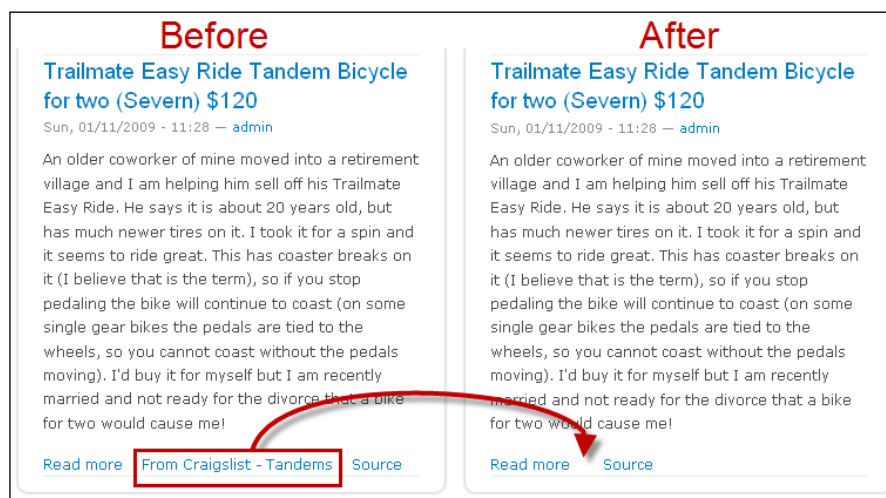
Recipe 71: Tweaking output with hook_link_alter

Ingredients
Recipe 70, completed to at least Step 8

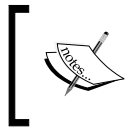
This recipe has two major goals. The first is to remove the link to the parent feed that appears under each node in *Recipe 70*. The second is to provide a slate for implementing the debugging strategies found in the next recipe, to help you to figure out how to make such tweaks on your own.

The **SimpleFeed** module creates several links in the `$links` array, which many people will find to be extraneous. The SimpleFeed documentation simply says "If you want to change or remove the links at the bottom of each feed node or feed item node, simply use Drupal's `hook_link_alter` to alter them", and it offers a link to the Drupal API page for the hook at http://api.drupal.org/api/function/hook_link_alter/5. Many Drupal users will feel a bit of a gulp in the throat when asked to implement such a hook. We actually need only a few lines of code.

Our link removal strategy has two basic steps: create a module, and enable it. The following image shows the results before and after our module is enabled:



1. In the `<DRUPALROOT>/sites/all/modules` directory, create a subdirectory named **custom**. Create a subdirectory of **custom** named **this_site**.



We could also choose to be more specific for the BikeHon site, calling directory bikehon. We'll use a generic example here, making sure that the directory name, and the upcoming .info and .module files all have matching names.

```
cd sites/all/modules
mkdir custom
cd custom
mkdir this_site
```

2. In the <DRUPALROOT>/sites/all/modules/custom/this_site directory, create a file named **this_site.info**, with the following text:

```
; $Id$
name = This Site
description = "Custom module for this site"
dependencies = simplefeed
package = Other
```

3. In the same directory, create a file named this_site.module, with the following code:

```
<?php
/**
 * Implementation of hook_link_alter().
 */
function this_site_link_alter(&$node, &$links) {
  # Enter the debug code of your choice here.
  # Be sure to remove the initial pound signs.

  // Remove simplefeed item parent link
  if ($node->type == "feed_item") {
    $links['simplefeed_item_parent'] = array();
  }
  // empties item parent array.
}
```

4. Enable the module, and visit the panel page bikes-for-sale. The feed item parent links are gone.



If you get a blank screen, go back and check your code. Make sure to check for closing parenthesis for each open one. Also, check your quotation marks to make sure they are simple quotation marks, and not fancy typographic ones.

Recipe notes

- Our little module empties the `$links['simplefeed_item_parent']` array, removing the link that would otherwise appear below the node.
- Note that there is no closing PHP tag. This is considered a best practice for modules. Here are some guides for when to use (or not use) surrounding PHP tags.

<code>.info</code> files	No beginning or ending PHP tag
CCK Allowed Values	No beginning or ending PHP tag
Computed Field module field	No beginning or ending PHP tag
Views Argument Handling Code	No beginning or ending PHP tag
<code>.module</code> files	Include beginning <code><?php</code> but not ending tag.
<code>template.php</code>	Include beginning <code><?php</code> but not ending tag.
<code>.tpl.php</code> files	Surround individual variables with PHP tags: <pre><?php print \$variablename; ?></pre> PHP tags may also be used to surround "if" statements, using a special syntax. <pre><?php if (\$taxonomy): ?> <div class="terms"> <?php print \$terms ?> </div> <?php endif;?></pre>
Block	Include beginning and ending <code><?php ?></code> tags when the PHP code input filter is selected.
Content Body	Include beginning and ending <code><?php ?></code> tags when the PHP code input filter is selected.
Panels custom body content	Include beginning and ending <code><?php ?></code> tags when the PHP code input filter is selected.
Views Custom field value	Include beginning and ending <code><?php ?></code> tags
Views Headers, Footers, and Empty text , for Pages and Blocks	Include beginning and ending <code><?php ?></code> tags, when the PHP code input filter is selected.

Paranoia?



If all the places for entering PHP seem a little scary — from a security perspective — consider the paranoia module, <http://drupal.org/project/paranoia>. Paranoia (the module) may not be a bad idea. If you have multiple administrators, you may disable PHP access for anyone but the super-administrator.

Recipe 72: Debugging

Ingredients

Devel module: <http://drupal.org/project/devel>

In this special set of recipe notes we look at a variety of ways to determine what's in a variable such as `$node`. This proved vital information to help determine the name for the `$links['simplefeed_item_parent']` array element, for instance, above. We do not have to divine this information!

Experiment with as many of the approaches as suit your fancy. You are sure to develop some favorite functions in the mix (the author's three favorites are `dpr()` — which requires the **Devel** module, `mail()` — when it's helpful to have a separate window for viewing the variable contents, and the **Drupal For Firebug** module).

The following debugging statements can be placed directly in the module code in *Recipe 71*, replacing the lines with the pound signs. They can also be widely used in modules and template files throughout Drupal. The statements are appropriate for use in development servers, rather than live sites.

These debugging statements work with core Drupal.

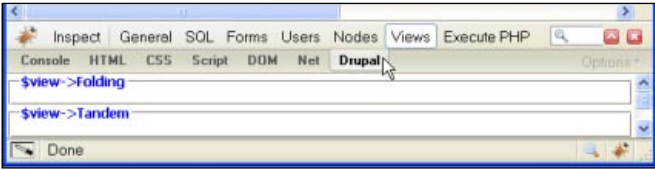
Statement	Description
<pre>drupal_set_message('<pre>' . print_r(\$node, TRUE) . '</pre>');</pre>	<code>print_r()</code> is a standard PHP function to return an array in human-readable form. The surrounding <code><pre></code> tags retain the helpful indented formatting when displayed in a browser.
<pre>drupal_set_message('type:' . print_r(\$node->type, TRUE));</pre>	You may <code>print_r()</code> any array or an individual array element.
<pre>mail("youremail@yourdomain. com", "[debug] ", print_ r(\$node,true));</pre>	This emails you the error message if your system is set up to send mail.

Statement	Description
<pre>print '<pre>'; print htmlspecialchars(print_r(get_defined_vars(), TRUE), ENT_QUOTES); print '</pre>';</pre>	This pure PHP code will print many more available variables. It generates a lot to look at, but it will reveal variables that don't show up when just examining an individual variable such as <code>\$node</code> .
<pre>print '<pre>'; print print_r(\$GLOBALS); print '</pre>';</pre>	This pure PHP code will print all of the global variables. It's easy to get more than 100,000 lines in the output. Occasionally helpful, but be prepared to wait.
<pre>print '<pre>'; print_r(\$variables); print '</pre>';</pre>	The <code>\$variables</code> variable will be empty inside a module, but proves very useful inside template (<code>tpl.php</code>) files.

These following debugging statements rely on the **Devel** module. They tend to be quicker to type, and so easier to use.

<code>dprint_r (\$node)</code>	This is similar to <code>print_r()</code> , but saves the trouble of having to type the <code></pre></code> tags, and it only prints for those who have the 'access devel information' permission.
<pre>dpr(\$node); dpr(\$node, FALSE, __FUNCTION__ . ': ');</pre>	dpr() is shorthand for dprint_r() An example of the dpr() function with parameters to print the source function name followed by a new line. Note: the statement is purposefully split across two lines, to create the new line in the output. This works because selected dpr() parameters are enclosed by HTML <code><pre></code> tags.
<code>dpm(\$node);</code>	Print a variable to the 'message' area of the page.
<code>dsm(\$node);</code>	This function is the same as dpm() , but probably easier to remember, since the original core Drupal function is named drupal_set_message() .
<code>dvr(\$node);</code>	This function is like <code>dpr</code> , but uses var_dump() instead. var_dump() is a PHP function, which includes the sizes of arrays and variables.
<code>dvm(\$node);</code>	var_dump() a variable to the 'message' area of the page. Uses drupal_set_message()

Devel is the most commonly used debugging module, but the following resources also offer debugging support.

Resource	Description
http://drupal.org/project/devinfo	DevInfo: A nice utility, to be used only on a development server, since it offers no permissions for concealing development data by role.
http://drupal.org/project/drupalforfirebug https://addons.mozilla.org/en-US/firefox/addon/1843 https://addons.mozilla.org/en-US/firefox/addon/8370	Drupal for Firebug requires the installation of two Firefox add-ons (Firebug, and Drupal for Firebug), along with the Drupal for Firebug module. Well worth installing, it sends Drupal debugging and status messages to Firebug.
	
http://drupal.org/project/trace	Trace
http://www.johnandcailin.com/blog/cailin/log4drupal-logging-api-drupal	Log4Drupal

Drupal is full of objects and arrays. In fact, many Drupal object variables contain arrays. Debugging strategies help sort them out.

The `<PRE>` tag preserves line breaks and indents, making printed objects and arrays much easier to understand.

Recipe notes

- If you wish to explore the Devel functions in more detail, see http://api.freestylesystems.co.uk/api/function/dprint_r/5/references.
- Both Devel and the Drupal for Firebug modules offer access permissions, at `admin/user/access`.
- From our first debugging statement, we may see large photographs in the feed. `htmlspecialchars()` enclosed in a `<PRE>` tags (the fifth statement in the tables above) will come close to "replicating" the deprecated tag `<XMP>` tag, enabling us to see the HTML for each image, rather than the image itself. Yes, the `<XMP>` tag is deprecated, but for quick debugging, sometimes it can be easier to remember and type. For instance: `drupal_set_message('<xmp>' . print_r($node, TRUE) . '</xmp>');`

Recipe 73: Node-feed-item.tpl.php

Ingredients

Devel module: <http://drupal.org/project/devel>
Completed Recipes 70 and 71

Let's say that instead of removing a single link from the `$links` array, as we did in *Recipe 71*, that we prefer to remove all of the links under the node. That is, we'll remove comments (if they were enabled), **Read more**, feed parent item, and the **Source** link. We just want the links gone, and we want the title, instead, to link directly to the craigslist post at craigslist.org (we'll keep taxonomy links if we have taxonomy terms).

1. Install and enable the **Devel** module.
2. Go to the theme directory, usually `<DRUPALROOT>/sites/all/themes/<THEMENAME>`. Copy `node.tpl.php` to `node-feed_item.tpl.php` (note: `feed_item` is the name of our content type).
3. Delete the part of the template file that prints the links. In most template files it will look like this:

```
<?php if ($links): ?>
  <div class="links">
    <?php print $links; ?>
  </div>
<?php endif; ?>
```

4. Add a variable debug line to the file such as:

```
dpr($node);
```

(Remember that this function is dependent on the **Devel** module. See *Recipe 72* for additional debugging options.)

5. Visit one of the feed items and explore the node object. Note that the element for the source craigslist URL is named **url**.

```
[url] => http://baltimore.craigslist.org/bik/1015073854.html
```

In `node-feed_item.tpl.php`, change the `$node->node_url` variable name (which of course, is the URL for the Drupal node) to `$node->url`.

6. Remove the debug line that we created in Step 4.
7. Visit the node page, or the full panels page. The links under the node are gone, and the title links to the source URL on craigslist.

If you should get a white screen instead of the intended effect, correct any missing characters, or check the site's log file for a helpful error message.

Recipe notes

- The source craigslist URL is also available at `$node->links['simplefeed_item_url']['href']` but `$node->url` is easier to type.
- Just as we can create `node-<CONTENTTYPE>.tpl.php`, we may also create template files such as:
 - `block-<MODULENAME>.tpl.php`
 - `block-<MODULENAME-DELTA>.tpl.php`
 - `block-<REGION>.tpl.php`
 - `block.tpl.php`

Recipe 74: Format the Pager with CSS-only modifications

Ingredients
A view with a pager

Over the next several recipes we will look at ways to customize the output of theme functions. In this recipe, we use CSS to alter the style of the site's pagers. The pagers are generated by the `theme_pager` function in the `includes/pager.inc` file.

In the standard default **Garland** theme, the pager looks like this (for a two-page view). The screenshot includes the pop up title that appears when mousing over a page number.



We will modify the pager to look like this:



1. Load a view that has a pager and examine the underlying source code in the browser. Note that in this example, page 1 is enclosed by the `<strong>` tag, which has a CSS class of `pager-current`.

```
<div class="pager">
  <span class="pager-list">
    <strong class="pager-current">1</strong>
    <a title="Go to page 2" class="pager-last active"
      href="/frontpage?page=1">2</a>
  </span>
  <a class="pager-last active" href="/frontpage?page=1">next ></a>
  <a class="pager-last active" href="/frontpage?page=1">last »</a>
</div>
```

2. Set the administrative theme to **Garland** at `admin/settings/admin`.
3. Download and untar a theme such as **hunchbaque** <http://drupal.org/project/hunchbaque> into `<DRUPALROOT>/sites/all/themes/` (do not install themes into the core `/themes` directory). **Enable** the theme, and set it as the default at `http://YOURSITE.com/admin/build/themes`.
4. Add the following CSS to the theme's `style.css` file (for the **hunchbaque** theme, this file is located in `<DRUPALROOT>/sites/all/themes/hunchbaque`).

```
.pager {
  margin: 1em 0;
}
.pager a:link, .pager a:visited, .pager strong.pager-current {
  background-color:#f5f5f5; /* Light gray */
  border: 1px solid #ccc; /* Medium Dark Gray */
  color: #2f4f4f; /* Slate Gray */
  font-size: .85em;
  font-weight: bold;
  margin: .3em .2em;
  padding: .3em .5em;
  text-decoration: none; /* Removes link underlines */
}
.pager a:hover, .pager a:active {
  border: 1px solid #3f5f99; /*Blue */
}
.pager strong.pager-current {
  background-color: #3f5f99; /*Blue */
  border: 1px solid #3f5f99; /*Blue */
  color: #fff; /*White */
}
```

5. Go to a multi-page view on the site, and try out the newly styled pager.

Recipe notes

- It pays to have a firm foundation in CSS. Get to know the available selectors, properties, and values. Understand the box model, floating, and positioning.
- CSS is powerful, but it can also be frustrating sometimes, given the challenges presented by multiple browsers. Two resources to help iron out puzzling cross-browser issues include: <http://www.positioniseverything.net/> and http://www.patriciarees.com/art198/week14/CMX_CSS_cheatsheet.pdf.
- Good web search terms include:
 - CSS syllabus
 - CSS cheatsheet
 - CSS cheat sheet
 - CSS hack
 - CSS tricks
 - CSS tips
- A particularly helpful "Cheat Sheet" is the Visibone.com unlaminated *Designer's Book* (for additional PHP, MySQL, and JavaScript information, get Visibone's *The Everything Book*).
- Consider also installing the Colorzilla add-on <http://colorzilla.com/firefox/> to help with color selection.
- In addition to the CSS that we placed in the theme, Drupal core contains some default CSS. For the pager classes, this code—copied below from the `/modules/system/system.css` file—ensures that the pager is centered on a line by itself, with a half em of padding. Default CSS rules may be overridden in your theme.

```
.pager {
  clear: both;
  text-align: center;
}
.pager a, .pager strong.pager-current {
  padding: 0.5em;
}
```
- The Zen theme (<http://drupal.org/project/zen>) offers a helpful reference file which includes all of Drupal's core CSS. Look for `drupal5-reference.css` in the theme's main directory (this file will be helpful whether or not you use the Zen theme for your site).

Recipe 75: Overriding the theme_pager function

Ingredients

A custom theme (or copy Garland into sites/all/themes directory)

In *Recipe 74*, we styled the pager but we could not change the HTML markup with CSS. In this recipe we perform an override of the `theme_pager` function to capitalize the words **First**, **Previous**, **Next**, and **Last**.

1. Optional step: If you're using the good-old default Garland theme, be sure to copy the whole garland directory from `<DRUPALROOT>/sites/themes/garland` to a subdirectory of `<DRUPALROOT>/sites/all/themes/`.

```
cd <DRUPALROOT>/sites/all/themes/
cp -r ../../../../themes/garland/ ./garland-new
```

Enable and make this new theme the default at `http://YOURSITE.com/admin/build/themes`.
2. Open the `<DRUPALROOT>/include/pager.inc` file, and locate the `theme_pager` function. Copy the function to the clipboard (an alternative location from which to copy the file is `http://api.drupal.org/api/5/function/theme_pager`).
3. Paste the function at the bottom of the theme's `template.php` file (this file is usually found in the `<DRUPALROOT>/sites/all/themes/<THEMENAME>` directory).
4. Change the function name from `theme_pager` to `phptemplate_pager`.
5. Review the function, and note that it is actually concatenating the output of five other related theme functions (`pager_first`, `pager_previous`, and so on). The concatenation is performed using the dot equals (`'='`) assignment operator. Locate the words `first`, `previous`, `next`, and `last`. Change them to the proper case, i.e. change the first letter to a capital letter.
6. Visit a multi-page view on the site. The pager now uses capital letters (the appearance, of course, will depend on the style sheet).



7. It's good practice to place documentation above any theme override, including the API page. If the function you are overriding is not part of the core, see if you can find it on the Contributed Modules API page instead at <http://api.freestylesystems.co.uk/>. See *Recipe 92* for details. Go back to the `template.php` file and add the following above the function.

```
/**
 * Override theme_pager
 * http://api.drupal.org/api/5/function/theme_pager
 *
 */
```

Recipe notes

- The steps above can be summarized as follows: to override a theme function, copy the theme function to `template.php`, rename it (change `theme_` to `phptemplate_`), and modify.
- When copying and pasting text, special characters can sometimes display in an odd and annoying way. For instance, a period — or even a 'â' character — might appear where the < character to the left of the Previous text belongs in the pager. Changing to the UTF-8 character set will usually help. For instance, someone using the **vim** editor with the **putty** ssh client would implement the following settings:

In vim:

```
:set encoding=utf-8
```

In putty:

Right-click the **Title Bar** | **Change Settings** | **Window** | **Translation** | **Received data assumed to be in which character set** | **utf-8**

- The theme function may be used for view-specific code. Add `global $current_view;` to the top of the function to provide access to all the views variables. Then you may perform logic such as:

```
// sample view_name: frontpage
if($current_view->name == 'view_name' {
    Do something;
}
```
- There's a relatively new **Pager Reference** module available: http://drupal.org/project/pager_preference which allows users to selectively **Show 25/50/100**. This is a nice usability enhancement. The module requires the patching of `pager.inc` (see *Recipe 90*), and comes with a theme function so that you may override the page count settings.

Recipe 76: Understanding how Theme functions are called

As our `theme_pager` function demonstrates, it is not unusual to find themeable functions calling other themeable functions. Being able to detect these building blocks is a good step toward deeper Drupal understanding.

Themeable functions are called by modules and themes using the `theme()` function. Some examples will be helpful:

General Example

`theme_function_name` is called using the following syntax:

```
theme('function_name', parameter, parameter...)
```

Specific Example #1

1. Visit http://api.freestylesystems.co.uk/api/function/theme_views_view/5

This is an excerpt of the `views.module`, containing the **`theme_views_view`** function. Each time a view is built this function is run.

2. Locate the spot in the code where the **`theme_view_more`** function is called.
`theme('views_more', $view->real_url);`

(In the full code, the function result is assigned to the `$output` variable.)

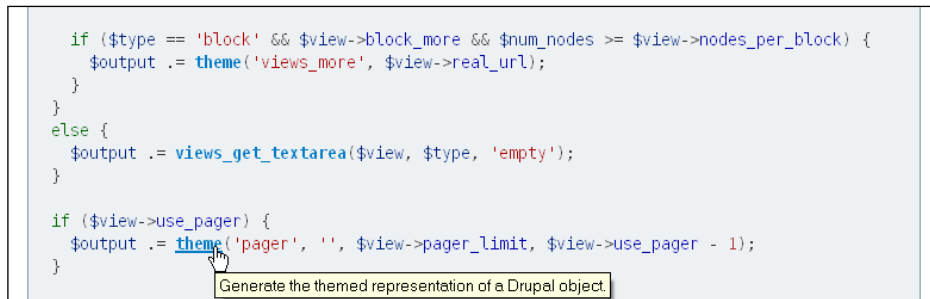
Make note of the surrounding "if statement" controlling the **`theme_views_more`** output.

```
if ($type == 'block' && $view->block_more && $num_nodes >=
    $view->nodes_per_block) {
    $output .= theme('views_more', $view->real_url);
}
```

The logic ensures that the **more** link appears only when the view type is a block, and only if the **[More] Link?** box is checked and the view has more items than the number of items already displayed in the block.

Specific Example #2: Calling themeable functions with multiple parameters

3. **theme_views_view** also calls the **theme_pager** function that we looked at in the three previous recipes. This places the pager at the bottom of every paged view. On the API page from Step 1, locate the spot in the **theme_views_view** code where the pager theme function is called. Most non-coders will give this code a puzzled look, so don't worry if you feel puzzled. We will next look at steps you can take to help you to decode the detail.



```
if ($type == 'block' && $view->block_more && $num_nodes >= $view->nodes_per_block) {
  $output .= theme('views_more', $view->real_url);
}
else {
  $output .= views_get_textarea($view, $type, 'empty');
}

if ($view->use_pager) {
  $output .= theme('pager', '', $view->pager_limit, $view->use_pager - 1);
}
```

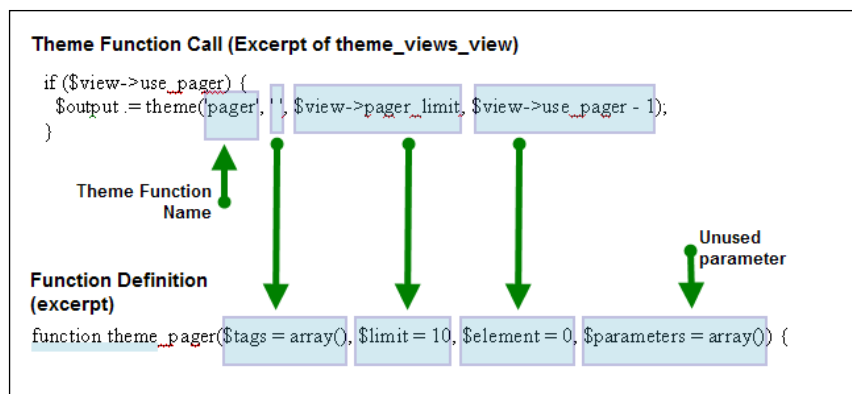
Generate the themed representation of a Drupal object

4. Remember the first parameter (in this case, **pager**). Make a mental note that the theme function you'll be looking for is **theme_pager**.
5. Click on the **theme** link.
6. On the API page, click on the **List References** link. This brings you to <http://api.freestylesystems.co.uk/api/function/theme/5/references> and shows every theme function available (limited to the modules installed on freestylesystems.co.uk site).

Alternatively, if you're at <http://api.drupal.org/api/function/theme/5>—on the API site for core Drupal—scroll down to the link named **201 functions call theme()**, and click on it.
7. Do a Find (*Ctrl+F* in many browsers) on the page for **theme_pager**.
8. Scroll down, and click on the **theme_pager** link, and read up on the four parameters.

Recipe notes

- The figure below shows how the calling function in an excerpt of **theme_views_view** maps to the **theme_pager** function:



- See *Recipe 92* for a browser Search Plugin for the Contributions API site.
- One attitude to adopt when looking at API detail pages can be "that's interesting, but I don't need to use that now". For instance, it may be interesting to note that an **\$element** parameter is available which is "an optional integer to distinguish between multiple pagers on one page". We were able to override this theme function without knowing this, but it's good to keep in the back of your mind for the time when a use case might arise for multiple pagers.

This table lists selected theme functions that call other theme functions:

This theme function...	calls this or these theme function(s)
theme_blocks	theme_block
theme_comment	theme_mark
	theme_username
	theme_links
theme_feed_icon	theme_image
theme_node	theme_username
	theme_links
theme_node_list	theme_item_list
theme_page	theme_blocks
	theme_breadcrumb
	theme_menu_local_tasks
	theme_help
	theme_status_messages
	theme_closure

This theme function...	calls this or these theme function(s)
theme_pager	theme_pager_first theme_pager_previous theme_pager_list theme_pager_next theme_pager_last
theme_table_sort_image	theme_image
theme_user_picture	theme_image
theme_views_bonus_export_doc	theme_views_view_table
theme_views_bonus_export_text	theme_views_bonus_export_txt_ separator
theme_views_edit_view	theme_fieldset
theme_views_filters	theme_table
theme_views_view_list	theme_item_list
theme_views_summary	theme_item_list
theme_views_view_table	theme_table
theme_xml_icon	theme_image
themes_views_view	theme_pager theme_views_more

Recipe 77: Override theme_views_view

We've taken a look at some sections of the `theme_views_view` function in the previous recipe. Now let's take a closer look, to give us more power to control our view.

The `theme_views_view` function may be roughly divided into six sections:

1. Title and Breadcrumb (for pages).
2. Header.
3. Exposed Filters (for pages, not blocks).
4. Content, or Summary view, plus the Footer. Also, for blocks only: the More link (alternatively, if there's no output, print the empty text).
5. Pager (pagers are only available for page views).
6. The final section that prints the view to the page.

Our recipe steps now begin:

1. Go to http://api.freestylesystems.co.uk/api/function/theme_views_view/5 and print out the **theme_views_view** function (alternatively, you may print the function directly from the `views.module` file).
2. Using the following pages as a guide, identify the six different sections, and label them.

```

<?php
function theme_views_view($view, $type, $nodes, $level = NULL, $args = NULL) {
    $num_nodes = count($nodes);

    1 if ($type == 'page') {
        drupal_set_title(filter_xss_admin(views_get_title($view, 'page')));
        views_set_breadcrumb($view);
    }

    2 if ($num_nodes) {
        $output .= views_get_textarea($view, $type, 'header');
    }

    3 if ($type != 'block' && $view->exposed_filter) {
        $output .= views_theme('views_display_filters', $view);
    }

    4 $plugins = _views_get_style_plugins();
    $view_type = ($type == 'block') ? $view->block_type : $view->page_type;
    if ($num_nodes || $plugins[$view_type]['even_empty']) {
        if ($level != NULL) {
            $output .= "<div class='view-summary ". views_css_safe('view-summary-', $view->name)
            ."'>". views_theme($plugins[$view_type]['summary_theme'], $view, $type, $level, $nodes,
            $args) . "</div>";
        }
        else {
            $output .= "<div class='view-content ". views_css_safe('view-content-', $view->name)
            ."'>". views_theme($plugins[$view_type]['theme'], $view, $nodes, $type) . "</div>";
        }
        $output .= views_get_textarea($view, $type, 'footer');

        if ($type == 'block' && $view->block_more && $num_nodes >= $view->nodes_per_block) {
            $output .= theme('views_more', $view->real_url);
        }
    }
    else {
        $output .= views_get_textarea($view, $type, 'empty');
    }

    5 if ($view->use_pager) {
        $output .= theme('pager', '', $view->pager_limit, $view->use_pager - 1);
    }

    6 if ($output) {
        $output = "<div class='view ". views_css_safe('view-', $view->name)
        ."'>$output</div>\n";
    }
    return $output;
}
?>

```

3. Once these separate sections are delineated, they may be shuffled around. Some common applications:

- Add an additional pager above the header
- Move the exposed filters above the header, so that people can see the filter search options near the top of the page
- Move the header above the title (perhaps the header contains a link to a pretty picture)

For now we will just add a second pager above the header. Copy the `theme_views_view` function to the `template.php` file. We know by now to change the name from `theme_views_view` to `phptemplate_views_view`. Additionally, we may target a specific view by appending a view name, for instance: `phptemplate_views_view_workshops`.

4. Copy the fifth section, the pager. Be sure to include the closing bracket. Paste just above the header section.

Recipe notes

- Be sure to always leave the sixth section as the last one.
- Programmers are not limited to shuffling the sections around, of course. Code within the sections may be modified as well.
- Just as we appended a specific view name to the `theme_views_view`, we may do the same for the style plugin (view type) theme functions: For instance, `phptemplate_views_view_table-<VIEWNAME>` overrides `theme_views_view_table`, and `phptemplate_views_view_list-<VIEWNAME>` overrides `theme_views_view_list`. We'll see the latter approach used by the Views Theme Wizard, in Recipes 79 and 81.
- If you get a white screen (sometimes referred to as the **White Screen of Death** or **WSOD**) when editing `template.php`, check that for each beginning bracket there is an end bracket, check the site error log, or restore your backup copy.

Recipe 78: Total Item count

You may add the total node count to the view title by performing a theme override of the `theme_views_view` function.

1. Copy the `theme_views_view` function into `template.php`.
2. Rename the function to `phptemplate_views_view_<VIEWNAME>`.
(Be sure to substitute the name of your view.)

3. Change this section, near the top of the function:

```
if ($type == 'page') {
  drupal_set_title(filter_xss_admin(views_get_title($view,
    'page')));
  views_set_breadcrumb($view);
}
```

To this:

```
if ($type == 'page') {
  drupal_set_title(filter_xss_admin(views_get_
    title($view, 'page')) . ' (' . $view->total_rows . ')');
  views_set_breadcrumb($view);
}
```

Recipe notes

- `$view->total_row` only exists if the view has a pager. If there is no pager, use `$view->num_rows`, instead.



The `$view->total_row` variable does not appear when we clone a view. So how do we know of its existence? One way is to temporarily add the following code to the Views header (be sure to set the input format to PHP).

```
<?php global $current_view; ?>
<pre><?php print_r($current_view) ?></pre>
```

This displays the full `$view` object! Review *Recipe 72* for more techniques for determining variable contents.

- Resources for other worthy approaches to adding totals to views are discussed at:
 - http://www.michaelanello.com/adding_node_counts_view_titles_views_1x (Discussion includes a helpful module)
 - <http://drupal.org/node/230240>
 - <http://drupal.org/node/131031>
 - **Views Calc** module, <http://drupal.org/node/336966> (See *Recipe 63*.)

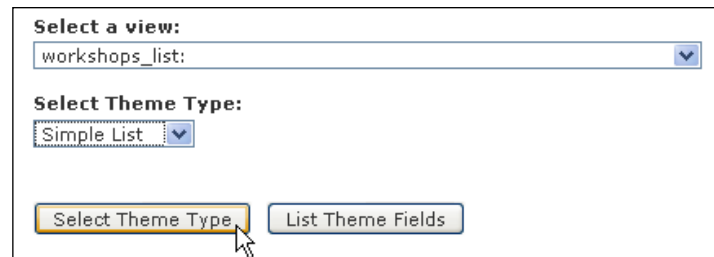
Recipe 79: Views Theme Wizard

Ingredients

Views theme wizard, comes with Views module
Completed Recipe 34, or view of your choice

The Views Theme Wizard has no magic wand, but it does offer a lot of power to help you to customize your views.

1. Enable the **Views Theme Wizard** module at `admin/build/modules`.
2. Clone a view such as `workshops`, from *Recipe 34*. Rename it to **workshops_list**, and change the URL to **workshops-list**. Change the View type to **List View**. Why change to a list? Because the Views Theme Wizard only works on **List View** types!
3. Visit the newly cloned view, and observe the starting point, before applying the wizard.
4. Visit `admin/build/views`, and select **Theme Wizard**.
5. Select a view, and the **Simple List** Theme type. Click **Select Theme Type**.



6. Three boxes of code appear on the resulting page that you can incorporate into your theme. Follow the instructions provided by each box.
 - Copy the full `phptemplate_views_view_list_workshops_list()` theme function (including the comments) to `template.php`
 - Create a new file in the themes directory called `views-list-workshops_list.tpl.php`, and paste the template code from the second box.
 - Create a new file in the themes directory called `views-list-workshops_list.css`, and paste the CSS from the third box.
7. Now we may manipulate the CSS and template file to your heart's content. Before doing that, let's pause for some questions and answers.

Frequently asked Questions:

Q: Wait a minute, I don't see anything. Why are there just empty HTML tags?

A: Make sure the file names were printed accurately. A template named `views-list_workshops_list.tpl.php` won't work. Neither will `views-list-workshops-list.tpl.php`. What we're after is precisely the name that is mentioned in the views template wizard.

Q: Where to start?

A: Start by reviewing the markup. Below is an excerpt from the markup code that we pasted into `views-list-workshops_list.tpl.php` (the generated markup is the same for all three theme wizard types). For each node, the whole set of DIVs will be surrounded by an unordered list.

```
<div class="view-label view-field-title">
  <?php print $title_label ?>
</div>

<div class="view-label view-field-title">
  <?php print $title?>
</div>

<div class="view-label view-field-field-workshop-date-value">
  <?php print $field_workshop_date_value_label ?>
</div>

<div class="view-field-view-data-field-workshop-date-value">
  <?php print $field_workshop_date_value?>
</div>

<div class="view-label-view-field-field-workshop-description-value">
  <?php print $field_workshop_description_value_label?>
</div>

<div class="view-field view-data-field-workshop-description-value">
  <?php print $field_workshop_description_value?>
</div>
```

Q: Where did all those variables come from?

A: Creating those variables is the job of the `phptemplate_views_view_list_workshops_list()` theme function that we pasted into `template.php`. (We're lucky that the theme wizard generated all that code for us!)

Q: I can clearly see that fields such as `$field_workshop_date_value_label` are supposed to print; So why is no label showing?

A: The field labels variables are populated when the view has a **Label** defined in the **Fields** fieldset.

Q: Alternatively, I can clearly see that I've set the fields to hidden, in admin/content/types/workshop/display, so why are they showing?

A: Our template file `views-list-workshops_list.tpl.php` is overriding those settings. If we want to be sure that the labels don't print (also not printing their surrounding divs), we can delete them from the `tpl.php` file.

Q: What about all those other variables listed in `views-list-workshops_list.tpl.php`?

A: They may be manually added. For instance, you could include `<?php print $stripe?>` to add the words "even" or "odd" to a class.

Q: Well, I certainly don't want that bullet in there. How can I remove it?

A: List items do not require bullets, just style them away. In most cases we can do something like this:

```
.view-workshops-list ul li {  
  list-style: none;  
  margin-left: 0px;  
}
```

However for a theme that has already styled the `.item-list ul li` selector, we need to override the theme's settings. The framework theme, for instance, uses a background property to place a bullet image. This selector has greater "specificity" to successfully clear the bullet.

```
.view-workshops-list .item-list ul li {  
  background: transparent;  
  margin-left: 0px;  
}
```

8. Armed with this information, let's go ahead and style the view, by editing `views-list-workshops_list.css`. For brevity's sake, we'll just show the selectors that are actually being styled.



It's fine to leave a few blank selectors in the CSS as they may come in handy at some point. You may also remove them for tiny increases in performance.

```
.view-data-title {
  margin: 1em 0 0 0; /* space above each title */
  font: bold 1.3em Arial, sans-serif;
}
.view-data-field-workshop-description-value {
  margin: 0 3em; /*Indent left and right margins */
  width: 20em;
  line-height 1.3em;
}
.view-data-field-workshop-description-value p {
  /* Reduce space between paragraphs */
  margin-bottom: .6em;
  line-height: 1.5em;
}
.view-data-field-workshop-description-value img {
  display:block;
}
/* images placed by IMCE in the content will get a line of their
own. */
  margin: 1em 0;
}
.view-workshops-list .item-list ul li {
  background:transparent;
}
/* Override the Framework theme's image bullets */
  margin-left:0px;
}
```

9. Finally, we'll add some quick logic into the template to create a default description if one hasn't been written yet.

```
<div class="view-field view-data-field-workshop-description-
value">
<?php if($field_workshop_description_value): ?>
  <?php print $field_workshop_description_value?>
<?php else: ?>
  <p>We've got a great program coming up. Check back here for
  details.
  </p>
<?php endif; ?>
</div>
```



This logic could also go into the theme function in `template.php`. This is a judgment call.

Labor Day Hike and Picnic

September 7, 2009

Meet at 11:30 at the Shady Park Pavilion.
Bring your favorite healthy dish, recipe card,
comfortable hiking shoes, musical
instruments, swimsuit, and game items
(volleyball, etc.)

Hikes (4 or 7 miles) are scheduled to leave
at 2pm. We have the Pavilion until 7 pm.

Pool is 10 minute walk from the Pavilion.

Rawsome Food Prep with Keisha

September 14, 2009



Keisha shares Luscious Lasagna with
cashew cheese, salad with herb dressing,
raw apple pie. Yum!

Medicinal Herbs

September 21, 2009

We've got a great program coming up.
Check back here for details.

Recipe notes

- If you are using the Zen theme, visit <http://drupal.org/node/220111> for a helpful fix to the Views Theme Wizard.

Handbook Page: How to Theme Views

I'd like to turn your attention to a particular Drupal handbook page: How to Theme Views: <http://drupal.org/node/128741>. If you are a coder—or an aspiring one—this is a must-read-and-try. It offers a special and flexible approach which builds on the Views Theme Wizard. We start with a view that contains only one field: the node ID. The template file then loads the full node, providing access to all its elements.

A few `.tpl.php` snippets may help you to get a better sense of this approach. The example comes from a site that streams both news and events to the home page.



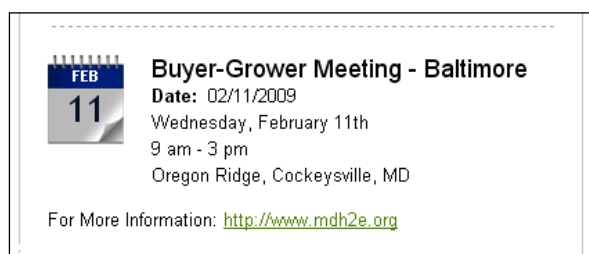
```
<?php
//Load the full node from the Node ID;
    $node = node_load($node->nid);
//Add the stylesheet; Note the helpful path_to_theme
//function;
    drupal_add_css(path_to_theme() . '/views-list-
frontpage.css');
?>
// The link below will actually resolve to the full
//path;
    <h2><?php print l($node-> title, "node/" .
$node->nid) ?></h2>
//print the teaser;
    <?php print $node->teaser ?>
// The CCK Place field is printed if the node is an
//event;
    <?php if ($node->type=="event": ?>
        <br />Place: <?php print
$node->field_place[0]['value'] ?>
    <?php endif; ?>
```

Recipe 80: Date badge

Ingredients:

CCK module: <http://drupal.org/project/cck>
Date module: <http://drupal.org/project/date>
Enable Date, Date API, Date Popup, and Date Timezone modules
Node Go To module: <http://drupal.org/project/nodegoto>
Recipe 34

In this recipe we create little calendars, or "date badges" that display the dates of upcoming events. The badges will display in nodes, teasers, and views.



This recipe begins in a fashion similar to *Recipe 34 (Block of Upcoming Workshops)*, with a couple of key differences:

- Our content type will be called **event**, rather than **workshop**.
- Our view will create a page, but not a block, and it will use the **Teaser List** view type, so no field selection is required.

With these differences in mind, let's proceed:

1. Go to `admin/settings/date-time`, and ensure that the default time zone is set.

Turn to *Recipe 34*, and replicate Steps 3-9, remembering to substitute the word **event** for **workshop** each time it appears in a URL or setting (for instance, create a **Content type** named **event**, a field named **event_date**, configure the field at `admin/content/types/event/fields/field_event_date` and so on). Be sure that when you create the date field, it is in the **Timestamp** format, using the **Text Field with jQuery pop-up calendar**.

2. Set the labels to **Hidden**, at `admin/content/types/event/display`.

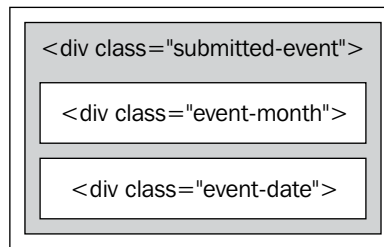
3. Create an **events** listing view.

Name	events
Description	Events Listing
Provide Page View	Checked
URL	events
View Type	Teaser List
Title	Upcoming Events
Use Pager	Checked
Nodes per Page	15
Filters	Node: Published Equals Yes Node: Type Is One Of Event Datestamp: event_date - Date (field_event_date) Operator and Option for the Datestamp: Set the Operator to greater than or equal to . Enter the word now in the Option field. Thus we are filtering to events that will be held in the future.
Sort	Datestamp: event_date (field_event_date)

4. Go to the <DRUPALROOT>/sites/all/themes/<MYTHEME> directory on the host system. Copy the node.tpl.php file from your theme to a new file named node-event.tpl.php. If it is like most themes, there is likely a section beginning with <?php if (\$submitted).
5. Replace that section of code with the following:

```
<?php if ($submitted): ?>
  <div class="submitted-event">
    <div class="event-month">
      <?php print format_date($node->field_event_date[0]['value'],
        'custom', "M"); ?>
    </div>
    <div class="event-date">
      <?php print format_date($node->field_event_date[0]['value'],
        'custom', "j"); ?>
    </div>
  </div>
<?php endif; ?>
```

This code signals that if an event is submitted, the template will print two `<div>` blocks nested inside a third **submitted-event** `<div>`. The **event-month** `<div>` is formatted to display the three-letter month abbreviation ("M"), while the **event-date** `<div>` is formatted to display the date, with no leading zeroes ("j"). A visual summary of the code is below.



6. We will now style these three classes, by adding the following CSS to the theme's `style.css` file:

```
.submitted-event {
    display: block;
    width: 55px; /* same width as the icon-calendar.png image */
    height: 64px;
    background: url(images/icon-calendar.png) no-repeat;
    float: left;
    margin: 0 20px 20px 0; /* right and bottom margin */
    text-align: center;
}

.event-month {
    color: #fff; /* white */
    text-transform: uppercase;
    font-family: Myriad, "Trebuchet MS", Arial, sans-serif;
    font-size: .85em;
    font-weight: bold;
    padding: 5px 0 0 0;
    /*remove background if images/icon-calendar.png is present */
    background: maroon;
}

.event-date {
    color: #403A5A; /* dark blue */
    font-family: Myriad, "Trebuchet MS", Arial, sans-serif;
    font-size: 2em;
    padding: 12 px 0 0 0;
    /*remove background if images/icon-calendar.png is present */
    background: gold;
}
```

Recipe notes

- To see some date badges in effect, visit <http://gestaltware.com/drupal>, <http://pingv.com/blog>, and <http://www.BaltimoreUrbanAg.org/>.
- In our style sheet, the **.submitted-event** class contains the background property that sets the URL of the underlying calendar image (Calendar graphic and CSS courtesy of TopNotchThemes.com).
- With some modification you may be able to use the free daily calendar icons available at: <http://www.mattvarone.com/icon/free-daily-calendar-icons-set-for-css/> or <http://www.mattvarone.com/icon/free-mini-calendar-icons-set/>.
- The underlying graphic is not actually required for an effective date badge. We could just set the backgrounds of the `<div>` blocks to solid colors (if you *do* have a background graphic, be sure to delete the background colors from **.event-month** and **.event-date**). Feel free to adjust the padding (or any other aspect of these styles) to get a satisfactory result.
- If the weekday is not included in a badge, it's a good idea to include it as text in the content description. With some attention, though, a third `<div>` could be added to incorporate the weekday into the badge. A check of <http://us3.php.net/date> shows that a lower-case "l" prints the full weekday ("Wednesday") while a capital "D" prints the three-letter abbreviation ("Wed").
- If an existing content type already has a date field of the **Date** type (ISO format), you will need to create a new **Datestamp** field, and then re-enter the dates into that field, before deleting the original field.
- Date badges may be used to display the date the content was posted, as well as CCK dates. The created date field is stored in timestamp format by default, so no special changes are required.

```
<?php print format_date($node->created, 'custom', "M"); ?>
<?php print format_date($node->created, 'custom', "j"); ?>
```
- The tutorial at <http://drupalsn.com/learn-drupal/drupal-tutorials/events-listing-system-using-date-cck-and-views-modules-part-3> implements the `format_date()` function to provide helpful filtering through argument links in a view header. It's worth checking out.
- See the handbook page at <http://drupal.org/node/92460> for general documentation on date field types and widgets.

Recipe 81: Grouped View

Ingredients

Completed Recipe 79, or thorough read-through

Completed Recipe 80, to at least Step 3

Computed field module: http://drupal.org/project/computed_field

Regenerate module: <http://drupal.org/node/195013#comment-854212>

Devel module: <http://drupal.org/project/devel>

The **Theme wizard**, which we used in *Recipe 79* for a **Simple List** view, also offers a **Grouped List** view. We'll group items in a view by month.



The most convenient method for grouping with the Theme wizard is to simply store a new variable in the database, containing the group name. We'll use the **Computed Field** module to achieve this.

1. Complete *Recipe 80*, up to at least Step 3.
2. Install and enable the **Computed Field** module.
3. Click on **Add field** at `admin/content/types/event/fields`. Name the field **Month**, and select the **Computed** field. Click on the **Create field** button.
4. Enter the following into the Computed Code area:

```
$node_field[0]['value'] = format_date($node->field_event_date[0]['value'], 'custom', "F");
```

You will probably recognize a portion of this code from *Recipe 80*. This time we're using the full name of the month (signalled by "F"), rather than the month abbreviation.

5. By default, the **Computed** field has the **Store using the database settings below** option checked. Leave that checked in order to make the field available to Views.
6. Set the **Length** to **15** (although for now we just need a month name, we're leaving the option open to have the **Month** say something like **November 2010**).
7. Click **Save field settings**.
8. View an event. Notice that the computed field does not display in a node by default. The field is available, though, and accessible to Views. If you wish to see the contents, one quick way is to click on the **Dev render** link, and examine the `field_month` array. At first, for existing content, the month value will be blank! You will need to save each existing node again to populate this field. You may do this manually (edit each content item and save), or you may install the regenerate module. Configure it to update **events** at `admin/settings/regenerate`.
9. Edit the **events** view. Change the **View Type** to a **List View**, and add the following three fields: **Node: Title**, **Datestamp: Date (field_event_date)**, and **Computed: Month (field_month)**
10. Go through the **Theme Wizard** again, this time, choosing **Grouped List** as the Theme Type, before clicking on **Select Theme Type**.
11. Select the **node_data_field_month_field_month_value** field on which to group, and click the **Generate Theme** button.
12. Copy the functions to their respective files.
13. Remove the Month divs (the label, and the value) from the `views-list-events.tpl.php` file. There is no need to print them, since the month is already displaying for each group.

Recipe notes

- It's probably best to filter the view for one-year time ranges. Otherwise, multiple years may end up under a common month header. Another option is to change the header field to the "F Y" custom format.
- The Regenerate module may be used to perform calculations across all existing nodes of content type. This should save you the trouble of having to manually save each node to update the computed field. The Regenerate module is not found on a regular project page, but rather in the Computed Field module issue queue.
- Review Chapter 4 for more tips on Date formatting. For instance, remember that you can easily set the `'date_format_medium'` as a variable override in `<DRUPALROOT>/sites/default/settings.php`.

Recipe 82: Body classes

"Body classes" might sound, at first, like a great way to lose weight and get strong, but in the context of a Drupal web site they are simply a set of CSS classes on the opening body tag. Examples might include:

```
<body class="sidebars">
<body class="sidebar-left">
<body class="front logged-in one-sidebar sidebar-left">
```

This may not look that exciting, but body classes actually yield a lot of power to Drupal themers.

For our example, let's consider a site set up by the designer to include a background image on two sidebar columns. The layout works for most of the content on the site. But we need to provide more space for selected wide pages. It's easy to configure blocks to show on every page except on the wide-stance view, for instance. That provides the physical room for the view. But what of the sidebar columns, which have a **repeat-y** image on the background property? Will the graphic still extend from top-to-bottom even if the sidebars are gone?

The good news for themes with body classes is that we can create a different appearance on pages that don't have sidebar blocks (or have just one sidebar block) just by using CSS. In this case, we use different graphics in the background property, depending on the number of sidebars. This CSS example is based on the **Zen** theme.

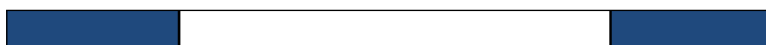
```
#main {
  width: 800px;
}

.two-sidebars #main-inner { /* two blue columns */
  background: white url(images/blue2cols.png) repeat-y scroll top left;
}

.sidebar-left #main-inner { /* Just one column, on the left */
  background: white url(images/blueleftcol.png) repeat-y scroll top left;
}

.no-sidebars #main-inner {
  background:white;
}
```

`blue2cols.png`, the image used when both sidebars are present, might look like the following image (though it could also be just 1-pixel high, since it repeats vertically).



Recipe notes

The table below includes seven themes that use Body classes. They are displayed in four groups due to similarities among some themes.

Garland, and Framework Themes	Blueprint Theme	Hunchbaque Theme	Zen, Basic, and Painted Wall Themes
sidebars	col-1	front	front
sidebar-left	col-2	not-front	not-front
sidebar-right	col-3	logged-in	logged-in
		not-logged-in	not-logged-in
		ntype-<TYPE>	node-type-<TYPE>
		sidebars	two-sidebars
		main-sidebar	one-sidebar sidebar left
		secondary-sidebar	one-sidebar sidebar-right
			no-sidebars
			page-<PATH>
			section-<SECTION>
			section-node-add
			section-node-edit
			section-node-delete
			section-node-<NUM>

Additionally, themes may includes *node classes* (such as `node-unpublished`, `node-mine`, `node-teaser`, `sticky`), *comment classes* (such as `comment-unpublished`, `comment-by-author`), and *block classes* (such as `Block-id`).

If you wish to incorporate classes into themes that don't already have them, this post will be helpful: <http://mydrupalblog.lhmdesign.com/classes-almost-everything-drupal-theme>.

Recipe 83: Carousel View

Ingredients

Completed Recipe 59 (Gallery)

Completed Recipe 60, Steps 1 and 2 (jQuery Update)

jCarousel: <http://drupal.org/project/jcarousel>

Carousel: <http://drupal.org/project/carousel>

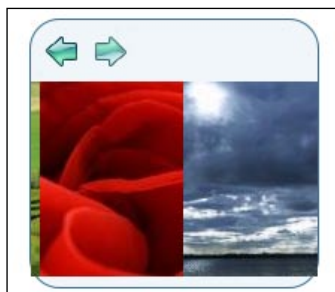
Enable the Views Carousel module

(Note: This recipe does not use <http://drupal.org/project/viewscarousel>)

This is something of an "uber-recipe" in terms of the sheer number of required modules. On the flip side, we won't ask for much exploration of code, as we have in some other recipes in this chapter. Once everything is installed, we're going to basically make a few Views changes, a quick theme function override, and a small change in the CSS.

1. Complete *Recipe 59* and the first two steps of *Recipe 60*.
2. Additionally, install and enable the **jCarousel** and **Views Carousel** modules using the URLs supplied in the ingredients section.
3. **Clone** the **Photos** view and change the name and URL to **slides**. Remove the **Title Field** and change the Photo **Option** to **130by130Square as link to node**.
4. Change the **View Type** to **Carousel View** and **Save** the view.

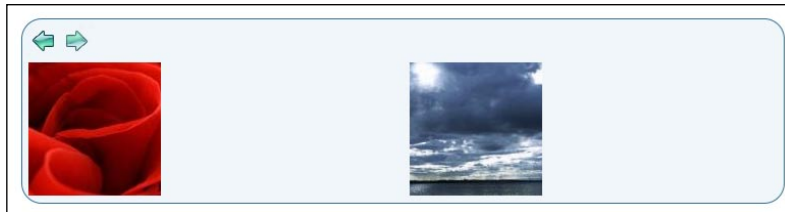
By default, the slides will look like this:



5. Copy the following two rules from the `skins/default/skin.css` file to your theme's style sheet. Change the width property in both rules from 16em to 100%.

```
.jcarousel-skin-default.jcarousel-container-horizontal {
    width: 100%;
    padding: 40px 0;
    padding-bottom: 0;
}

.jcarousel-skin-default .jcarousel-clip-horizontal {
    width: 100%;
    height: 100%;
}
```



6. In order to populate the carousel with more nodes, we will need to override the module's theme function. Copy the `theme_views_view_carousel` function to `template.php`, and rename the function to `phptemplate_views_view_carousel`. It's a lengthy function, but don't worry, we're just going to change one character:

Change:

```
'visible' => 2, // Visible items.
```

To:

```
'visible' => 5, // Visible items.
```

Our carousel now looks like this:



Recipe notes

- There are two different modules named Views Carousel. In this recipe we are using the **Views Carousel** module found at <http://drupal.org/project/carousel>. This module creates a subdirectory called `views_carousel` (the other Views Carousel module creates a directory called `viewscarousel`).
- A view named **carousel** seems to cause a SQL error with this our carousel module. Other standard view names work fine.
- If you prefer not to have an extremely long `template.php` file you may choose to replace the last step with an include file, as follows:
 - Copy the function to a file named `views_carousel_theme.inc`.
Rename `theme_views_view_carousel` to `phptemplate_views_view_carousel`, just as you would if it were part of `template.php`.
 - Add `<?php` to the top of the file (the closing php tag is not necessary).
 - In `template.php`, add the following:

```
/**
 * Include Files
 */
include_once(drupal_get_path('theme', theme_name) . '/views_carousel_theme.inc');
```

Be sure to substitute the name of your theme for `theme_name` above.

- It may be surprising to notice that the images used for the previous and next buttons (`prev-horizontal.png`, and `next-horizontal.png`) actually consist of four arrows, each.



Do a web search for **CSS Sprites** if you would like to learn more about this button rollover technique. The following sites will generate CSS sprites should you ever want to add them to your CSS bag of tricks:

- <http://www.csssprites.com/>
- <http://spritegen.website-performance.org/>
- You may choose to place your carousel in a block. Adjust the number of images for the available space in the theme override.

- There are actually a lot of nice solutions to draw from for rotating content. Included in the mix of Views-enabled solutions are: jCarousel Lite, Lightbox2 (described in *Recipe 60*), SlideShowPro Integration, Views Rotator, and Views Slideshow. Be on the lookout for upcoming Views integration with a module called Featured Content Slider. Other worthy solutions (without Views support) include Couloir Slideshow, Fading Slideshow, JQuery Slideshow, and Slideshow.
- The solution we provided offers a good example of overriding a stylesheets, overriding a theme function in an external file, and introducing the CSS Sprite technique.

Recipe 84: Views Dashboard—code approach

Ingredients

Path module (part of core)

Tabs module, part of JSTools module: <http://drupal.org/project/jstools>

Three completed views, without exposed tabs or special permissions

In this recipe we create a dashboard that provides quick access to several views created in this book. This involves some code, but it's fairly understandable, and easy to customize. In the next recipe, we'll use a code-free approach.

1. Enable the Path module if it is not already enabled.
2. Install the JSTools module, and enable the Tabs module.
3. Create a content **Page** called **Dashboard**, with a path of **dashboard**. Body text could be blank (or contain text, per your preference).
4. Go to `admin/build/blocks` and add a block called **dashboard**.
5. In a separate window, on the host system, open the `README.txt` file, located in `<DRUPALROOT>/sites/all/modules/jstools/tabs`. Copy the code appearing below the line that reads **Sample usage**. Paste the code into the Drupal block at `admin/build/block/add`.
6. Edit the code, until it reads as follows:

```
<?php
$view_calendar = views_build_view('embed',
views_get_view('calendar'), $view_args, true, 25);
$view_posters = views_build_view('embed',
views_get_view('posters'), $views_args, true, 30);
```

```
$view_timeline = views_build_view('embed', views_get_
view('timeline'), $views_args, true, 30);

$form = array();
$form['dashboard'] = array(
  '#type' => 'tabset',
);
$form['dashboard']['tab1'] = array(
  '#type' => 'tabpanel',
  '#title' => t('Calendar'),
  '#content' => $view_calendar,
);
$form['dashboard']['tab2'] = array(
  '#type' => 'tabpanel',
  '#title' => t('Posters'),
  '#content' => $view_posters,
);
$form['dashboard']['tab3'] = array(
  '#type' => 'tabpanel',
  '#title' => t('Timeline'),
  '#content' => $view_timeline,
);
print tabs_render($form);
?>
```

7. Make sure that the Input format is set to **PHP code**.
8. In the block configuration, set the visibility of the block to show only on the **dashboard** page.
9. Save the block and place it in the content region.
10. Visit your new dashboard page at <http://YOURSITE.com/dashboard>. Notice the new tabs. The appearance will vary based on your theme's style sheet.

Calendar	Posters	Timeline	
Poster	Year	Artist	
Don't Say You Didn't Know	1982	Iris Dishon	
Edward Said in Berkeley	2006	Jos Sances	
Israel/40th Anniversary	1988	Iris Dishon	
Unarmed Truth	2004	Jos Sances	

Recipe notes

- You may substitute available views in place of calendar, posters, and timeline. The four substitution areas will be in the initial variable names, the viewname argument for `views_get_view()`, and the `#title` and `#content` properties.
- Do not confuse the **JSTools Tabs** module with the **Views Tabs** module or **Quick Tabs**.
 - The **JSTools Tab** module creates two new form elements: `tabset`, and `tabpage`, which may be used in coding blocks or modules.
 - **Views Tabs** (not detailed in this book) uses Views filters to select nodes that will become individual tabs.
 - **Quick Tabs**—*Recipe 85*, offers a friendly interface for adding blocks and views to tabbed pages. In many cases, **Quick Tabs** will likely replace the need for the kind of manual coding covered in *Recipe 84*. One reason to use this recipe's programmatic approach is be to quickly enable your special theme's default tab style, if it's not already in the pick list of default styles (also, of course, code is easily integrated by programmers into larger modules).
- Neither tabbed dashboard approach is recommended for views with exposed filters.
- More information can be found on the **JSTools Tabs** approach at <http://drupal.org/node/124750>.

Recipe 85: Views Dashboard—module approach

Ingredients

Quick Tabs module: <http://drupal.org/project/quicktabs>
 Path module (part of core Drupal)
 Three views

1. Enable the path module if it is not already enabled.
2. Install and enable the Quick Tabs module.

3. Visit `admin/settings/quicktab` and select a tab style.

The screenshot shows the 'Quick Tabs Settings' interface. It features a 'Tab Style:' section with ten options, each with a radio button and a preview. The 'Default (no style)' option is selected. The preview for 'Default (no style)' shows two tabs labeled 'One' and 'Two'. Other styles include 'Mac' (rounded tabs), 'Zen' (flat tabs), 'Basic' (simple tabs), 'Metal' (metallic tabs), 'Phylactere' (speech bubble tabs), 'Arrows' (tabs with arrows), 'Garland' (garland-style tabs), 'Plastic' (plastic-style tabs), and 'Bullets' (bullet-style tabs).

4. **Save** the configuration.
5. Go to `admin/build/quicktabs` and read the descriptive text.
6. Click on **New QT Block**.
7. Create a block title: **Calendar/Posters/Timeline**.
Configure the tabs as follows:

Tab Title	Weight	Tab Type	Select a view	Limit	Build Type
Calendar	-10	view	calendar	50	embed
Posters	-9	view	posters	20	embed
Timeline	-8	view	timeline	30	embed

8. Create a content **Page** called **Tabs** and give it the pathname **tabs**. Body text could be blank (or contain text, per your preference).
9. Go to `admin/build/block`, and place the new block in a region. Then configure the block so that it appears only on the **tabs** page.
10. Visit `http://YOURSITE.com/tabs`.

Calendar Posters Timeline		
Poster	Year	Artist
Don't Say You Didn't Know	1982	Iris Dishon
Edward Said in Berkeley	2006	Jos Sances
Israel/40th Anniversary	1988	Iris Dishon
Rachel Corie	2004	Jos Sances

Recipe notes

- With a little bit of CSS attention, it is easy to create your own custom tab styles. Copy one of the existing tab styles to a new directory (being sure to copy the images subdirectory). Modify image files and CSS, and your custom tab styles will appear at `admin/settings/quicktab`.

Recipe 86: Great Online Theming resources

[**Ingredients**
A browser, curiosity, and a desire to learn]

There are a lot of online resources. Here is a selection of the best.

Theme Developer's Cheatsheet

www.minezone.org/blog/wp-content/uploads/2007/12/drupal-theme-developers-cheat-sheet.pdf

Theming handbook

<http://drupal.org/theme-guide/5>

Theme Snippets (part of theming handbook)

<http://drupal.org/node/45471>

How to define custom regions (part of theming handbook)

<http://drupal.org/node/29139>

Theming Views 1 (major section of the Views Documentation)

<http://drupal.org/node/42597>

How to theme Views (part of the Views Documentation, Tutorial Section)

<http://drupal.org/node/128741>

Theme development Support forum

<http://drupal.org/forum/3>

Validate CSS

<http://jigsaw.w3.org/css-validator/>

Combining different views using views offsets

<http://pingv.com/blog/slaug/200802/view-offsets>

Drupal Dojo

<http://drupaldojo.com/lesson/theming-like-a-pro>

<http://www.drupaldojo.com/lesson/theming-and-the-themer-pack>

Converting Mockups to DrupalThemes- DrupalCon presentation

<http://www.archive.org/details/DrupalconBoston2008-ConvertingMock-upsToDrupalThemes>

<http://tinyurl.com/c52zc3>. This is a short URL leading to a PDF from the presentation, on maureensmusings.com.

Theming Drupal

<http://11heavens.com/theming-Drupal> (\$25 Login required)

Nick Lewis' Drupal Hackers Cookbook

<http://www.nicklewis.org/drupal-hackers-cookbook>

Theme a field

<http://codepositive.com/theming-views-output-1-theming-title-using-function-views-theme-field>

Summary

We started this chapter exploring two ways to identify themeable functions on your site. First, we created a hyperlinked reference page, then we tracked the source names of themeable functions in the HTML source code.

We looked at a variety of approaches to control view appearance: CSS, theme overrides, template files, small custom modules, and the Views Theme Wizard.

Along the way, we were exposed to helpful code snippets for examining the contents of variables. Debugging code may be placed inside a view, inside a module, or inside various files that make up the theme.

We looked at ways to theme individual content types and individual views or view types. We spent a good deal of time looking at `theme_pager` and `theme_views_view`, exploring the use of the `theme()` function.

We had a chance to work with simplefeed, panels, a views carousel, and two ways to place views in tabs. We focused on the `datestamp` format for the CCK Date field, which eases special formatting for theming applications such as a date badge, or views grouped by month.

The chapter included special tables, covering when to use beginning and end PHP tags, which themes include body classes, and which theme functions call other theme functions.

We closed with some key resources that will help you in your work and play with Drupal Views theming.

8

Navigating the Online Drupal Community

Drupal.org is an amazing resource rich in conversations, modules, documentation, patches, and more. In this chapter, we cover some important skills for use on the hefty site: how to interact with the issue queue and how to create and apply patch files. We'll also look at some additional helpful online resources.

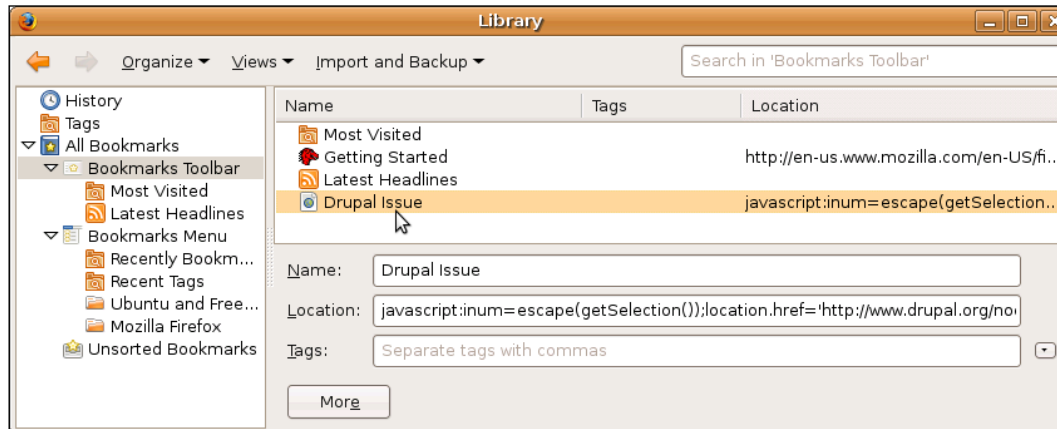
Recipe 87: Creating an issue Page Bookmark

[**Ingredients**
Web Browser]

The issue queue is the central place of progress for Drupal modules. It serves as a place to find answers, patches, new ideas, and work on common concerns. Issues are referenced by number. On occasion, a web page will contain an issue queue number in text form rather than a full link to the issue. This recipe, once set up, simply saves the trouble of having to type `drupal.org/node/<IssueNum>` into the browser address bar. Just select the number, and the bookmark will take you there.

1. In Firefox add a new Bookmark onto the toolbar. Select **Bookmarks** | **Organize Bookmarks** | **Bookmarks Toolbar** | **Organize** | **New Bookmark**

2. Set the **Name** to **Drupal Issue** and set the **Location** to the following:
`javascript:inum=escape(getSelection());location.href='http://www.drupal.org/node/'+inum.`



3. Visit a web page that contains an issue number and select the issue number text. For instance, try

`http://cvs.drupal.org/viewvc.py/drupal/contributions/modules/views/modules/views_taxonomy.inc.`

(Be sure to exclude the surrounding space and pound sign when selecting the number.)

Revision **1.2.4.84** - ([view](#)) ([download](#)) ([annotate](#)) - [[select for diffs](#)]
Thu Dec 4 01:47:51 2008 UTC (3 weeks, 6 days ago) by *sim*
Branch: **DRUPAL-5**
Changes since **1.2.4.83**: +1 -0 lines
Diff to [previous 1.2.4.83](#) , to [branch point 1.2](#) , to [next main 1.126](#)
[#142347](#) [by merlinofchaos: Fixed missing taxonomy terms in exposed filter.

4. Click the **Drupal Issue** button in the bookmark toolbar.

Recipe notes

- This bookmark approach may be replicated to visit a URL containing any selectable text. For instance, below is a variation to display all of your delicious bookmarks tagged with the selected text. (Delicious.com—also found at `http://del.icio.us`, is a wonderful online bookmark service.) Replace `<ACCOUNTNAME>` with your `delicious.com` account.

Name: **Delicious**

Location: `javascript:tag=escape(getSelection());location.href='http://delicious.com/<ACCOUNTNAME>/'+tag`

Recipe 88: Searching the Views issue queue

[**Ingredients**]
Web Browser

In this recipe we look closely at how to search the Views issue queue. The lessons apply to all other Drupal projects as well. It is always a good idea to search the issue queue for related content before posting.

1. Log on to `drupal.org` (if you are not already a member of the Drupal site, become a member).

Basic Search

2. Visit `http://drupal.org/project/issues/views`. At this main issue queue page you may search for text or filter by **Status**, **Priority**, **Category**, **Version**, or **Component**. These options are discussed in further detail below.
3. You may also sort the table of issues by clicking on the table header. By default, the table is sorted by date.

Advanced Search

4. Go to the Views issue queue Advanced Search page.

There are a variety of routes to get there:

- Visit the URL directly, at `http://drupal.org/project/issues/search/views`.
- From the project page (`drupal.org/project/views`), find the **Issues** block on the left, and click on the **Advanced Search** link.
- From the issue queue (`drupal.org/project/issues/views`), the **Advanced Search** Link appears under the title.

5. Get to know the search options.

Although there are ten form elements to choose, most users will routinely use just a few, leaving the other options blank.

- **Search For** (Routinely used): Enter search text. Use quotation marks to create a phrase.
- **Assigned:** This field is generally used by issue maintainers.
- **Submitted by:** This is most often used to find your own issues, though it could be used to see what other Drupal users are posting as well.
- **Participant:** This is also used to find your own posts. Note that **Submitted by** finds only the initial post by a user in the issue queue. **Participant** additionally includes responses to initial posts.
- **Status:** Leave blank to get all statuses. You may also select multiple options. For instance, you could select all issues designated as **needs work**, **needs review**, and **reviewed & tested by the community**. Scroll down the list and note Status filters such as **closed** issues, **duplicates**, issues that the maintainer **won't fix**, and features noted as **by design**. These are the statuses that are excluded if you select **-Open Issues-**.
- **Priority:** Leave blank to get all priorities.
- **Category:** Leave blank to get all categories.
- **Version** (Routinely used): A relative new option, **5.x issues** saves you the trouble of having to *Shift*+click on each Drupal 5 release name.
- **Component:** The views module issue queue offers more component options than most modules. As a result, users may not always be familiar with properly assigning a component when they create an issue. A search of **exposed filters** components, for instance, may not find as many results as a text search of "exposed filters." **Component** can occasionally be a helpful selection, but is most often left blank.
- **Issue Tags:** These may be a challenge to search since few people add tag issues. This may become a more popular option in the future.

Recipe notes

- Search ideas:
 - Find all your posts by filling in your `drupal.org` user name under participant.
 - Find patches by selecting all of the four patch statuses.
 - Find all documentation issues connected to Views for Drupal 5.x.
 - Go to another issue queue `http://drupal.org/project/issues/search/<MODULENAME>` and search for the word **Views**.

- From the module issue pages <http://drupal.org/project/issues/<MODULENAME>> you may also review module **Statistics**, and **Subscribe** to issues. Subscribe to your *Own Issues* (the default), *None*, or *All Issues*. I don't recommend the latter for the Views module as you will be setting yourself up for a deluge of email.
- Search across all projects at <http://drupal.org/search/issues>.

Recipe 89: Posting an issue

[Ingredients]
Web Browser

Posting a New issue

1. If you are new to posting Drupal issues, consider just reading the issue queue for at least several days before posting. This will help you to get a sense of the culture of issue queue interaction.
2. If you don't already have an account on drupal.org get one. Look for the User login block on the home page, and click on **Create new account**.
3. Complete the steps to login.
4. Search the issue queue before you post! (*Recipe 88*). If your topic already has an associated active issue, reply rather than posting a new issue.

Also, before posting to the issue queue in a panic read the Drupal **Troubleshooting FAQ** <http://drupal.org/Troubleshooting-FAQ>. For instance, standard fare is to increase memory in the face of the **White Screen of Death (WSOD)** or to disable buggy modules by setting the `status = 0` in the system table.

5. Be sure to know which version of the module you're using. Is it the *dev* (development) version? Is it the latest recommended release? The version number can generally be found at <http://YOURSITE.com/admin/build/modules>.
6. To start a new issue, go to <http://drupal.org/project/issues/<MODULENAME>> and click on **Create a new issue**. This directs the browser to: <http://drupal.org/node/add/project-issue/<MODULENAME>>.

For the Views module, the link at <http://drupal.org/node/add/project-issue/views> offers guidance (in bold!) for posting. Read it! Much of it applies to **Views 2** but it contains useful information for Views 1 users as well.

- Required fields for a new issue include **Version**, **Component**, **Category**, **Title**, and **Description**. Be thoughtful with these details. For instance, do not title your issue *HELP??!!* A much more useful description would be something like *Missing taxonomy terms in filters*.
- **Priority** should generally be left as **normal**. **Critical** is reserved for occasions when the module simply does not work.

Responding to an existing issue

You may also *respond to an existing issue* by selecting the **Add New Comment** link or one of the **Reply** links on an individual issue page. Another option is just to scroll down to the bottom of the issue page, and begin entering a response. Unlike some forum tools, in which replies are indented, all new comments are given a new comment number, and added to the bottom of the comments.

When responding to an issue you may take a variety of actions:

- Change the **Issue Title**. In general, don't change this unless you have a very good reason (for instance, if the original title is misleading, or spelled wrong). Some people are used to forums where a response can have a different name as the original post. In the issue queue, changing the name when responding to an issue actually changes the name of the issue. This is generally best left untouched.
- Change the **Project**. A question that someone asks in the Views issue queue may be more appropriately managed in the issue queue for a different module. This is a rare change generally left to the maintainer of one of the two modules who will know in which issue queue a discussion belongs.
- Change the **Version number**, **Component**, **Category**, or **Priority**. These changes are rare (correcting the version number is probably the most common). When changes are made, they are noted in the post as shown below:

Version:	5.x-1.6-beta5	» 5.x-1.6
----------	---------------	-----------

- Change **Assign**. Do not assign someone other than yourself to an issue. Assign yourself if you are sure that you will soon fix the issue. It is quite common to leave this as **Unassigned**.

- Change the **Status**. For instance:
 - Mark an issue as a duplicate (always provide a pointer to the issue it duplicates).
 - Note that a patch is reviewed and tested by the community
- Post a question, patch, answer, or idea related to the issue in the **Comment** section. Open the **Input format** fieldset below the comment field to see what markup is available. Note the `<code>` tag, for instance (and remember to close it with a `</code>` tag).
- Attach a file.

Recipe notes

- Remember that respondents and maintainers are volunteers. They are generally very busy people who want to help, but they do not have time to do free consulting.
- See the following pages for spirited discussions about issue queue etiquette:
 - <http://acko.net/blog/whats-wrong-with-drupal>
 - http://paul.leafish.co.uk/articles/drupal/on_subscribing_to_module_portingupdating_issues
- One discussion theme is the merit of simply sending the word **subscribe** to the issue queue. People sometimes do this so that they can track an issue—receiving an email alert each time something new is posted. On `drupal.org` it is possible to subscribe to a node only if you leave a comment, but most people prefer comments with substance.

You may create functionality similar to the Drupal issue queue on your own site by installing the **project**, **project_issue**, and **comment_upload** modules.

Recipe 90: Applying a Patch

[**Ingredients**]
Patch utility, or Eclipse IDE

A Drupal patch is a text file containing an update to Drupal code. Patches may include bug fixes or additional features. One great thing about Drupal patches is that the special text format makes code changes easy to read. Another is that patches are easy to install—far less time-consuming or error-prone than having to manually update code.

The following is a screenshot of a simple patch to correct spelling. Minus signs signal the original file (either a backed-up file or the original file located on CVS). Plus signs signal the edits made to the new file. Surrounding code is also provided for context.

```
--- gmaps.module.backup 2008-12-04 17:24:29.000000000 -0500
+++ gmaps.module        2008-12-04 17:34:43.000000000 -0500
@@ -392,7 +392,7 @@ function _gmaps_latlon_expand($element)
     'latitude' => t('Lat.'),
     'longitude' => t('Lng.'),
     'privacy' => t('Privacy'),
-    'add_marker' => t('You need to doubble click on the map to add coordinates.'),
+    'add_marker' => t('You need to double click on the map to add coordinates.'),
     'move_marker' => t('You can drag the marker to arrange the coordinates.'),
   );
```

The steps below are generic, but each step is followed by a specific detailed example. We will find and apply a patch that enables email field sorting and filtering.

1. Identify the patch in the issue queue and copy its URL to the clipboard. There is a bit of an art to this. Use the issue queue Search (*Recipe 88*) wisely and pay attention to feedback on the patch from others in the issue queue. Usually the most recent patch in the queue is the best one to try.
 - Visit the issue search for the email module at <http://drupal.org/project/issues/search/email>.
 - Search for the word **Views**. Select **- 5.x issues** and the four patch statuses (**needs work**, **needs review**, **reviewed and tested by the community**, and **patch (to be ported)**) and press **Search**.
 - Click on the title **Email not available as filter in Views**.
 - Click on the **Most recent attachment** link at the top of the issue page (this link is not present if the patch is in the initial post and no new patches have been uploaded).
 - Click on the **email.module.patch** link.
 - Copy the URL for the patch to the clipboard. In this case, this URL will be http://drupal.org/files/issues/email.module_5.patch
2. Open a window to the host account, and navigate to the module directory.
`cd <DRUPALROOT>sites/all/modules/email`
3. Backup the file that you intend to patch.
`cp email.module email.module.backup`
(This step is not entirely necessary, since the `-b` switch on the patch command will also do this. The author's preference is to make a manual backup.)

4. Get the patch file.

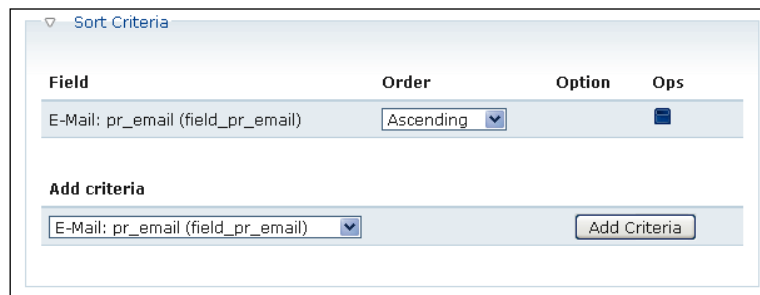
```
wget http://drupal.org/files/issues/email.module_5.patch
```

(There is no need to type the URL. Just use the paste function, which on a unix terminal is often the right mouse button.)

5. Patch the file.

```
patch < email.module_5.patch
```

6. If you have enabled the email module and created an email field, the email field now appears in the filter and sort fieldsets at `admin/build/views/add`.



Recipe notes

- Congratulations for trying this recipe! You are greatly enriching your Drupal experience by being able to apply patches to contributed modules. The recipe may look more daunting than it really is. We spent most of the recipe tracking down the patch, and placing it in the appropriate subdirectory. The actual patching part is a simple one-line command shown in Step 5.
- The steps for applying patches to Drupal core code (as opposed to contributed modules) are slightly different. Contributed modules use a command like this and run from the module subdirectory:

```
patch < file.patch
```

Drupal core patches, however, are applied from the Drupal root directory as follows:

```
patch -p0 < path/file.patch
```

- Users of the Eclipse editor are in luck with a quick patching feature. Select the **Project | Apply Patch** menu option in the **Team Synchronizing** perspective. The patch may be run from the clipboard, as well as from the patch file path.
- A quick way to see some of the recently run patch commands is to search the history file:

```
history | grep -i "patch"
```

- The easiest way to reverse a patch is to restore a copy of your backup. You may also use the capital `R` switch on the patch command to reverse a patch:
`patch -R <file.patch`
- Dmitri Gaskin offers a script for creating and reviewing patches
<http://dmitrizon.com/blog/blog/patch-creating-and-reviewing-script>.
- More detail on patching is available at <http://drupal.org/patch/apply> (handbook page) and <http://drupal.org/node/128209> (link to video on Drupal core patches).
- If you wish to include the author email in views (along with a number of other user-related fields, patch the `views_user.inc` with the patch at http://drupal.org/files/issues/views_20070511_user_table.patch. The file is located in the modules subdirectory, usually at `sites/all/modules/views/modules`.
- *Appendix G* contains a number of very useful patches to views and some related modules, sorted by type.

Recipe 91: Creating a Patch

[Ingredients]
Diff utility

Regardless of your Drupal programming skill level, you may submit patches to the Drupal issue queue. Module maintainers and users will appreciate patches ranging from spelling corrections, to more complex bug fixes and new features. Patches may be created with the `diff` command or the CVS `diff` command. The example below uses `diff`.

1. Backup the file intended for editing.
`cp views_ui.module views_ui.module.backup`
2. Edit the file with the editor of your choice.
`vim views_ui.module`
3. Run the `diff` command to create a patch. Notice that the command uses the `-u` and `-p` switches and compares the original file (first) to the edited file (second). The `diff` is then redirected to an output file, usually named with a `.patch` extension.
`diff -up views_ui.module.backup views_ui.module > views_ui.module.patch`

Patch command switches



The `-u` switch in the patch command stands for **unified context** diff, which is the desired format for Drupal patches. In addition to using the plus and minus characters to signal file changes and including surrounding code for context, this format also features informative double **at** signs. These enclose information about where the changes were made.

This line notes that beginning at line 858 in the original file, and at line 858 in the new file, a range of 7 lines is included.

```
@@ -858,7 +858,7 @@
```

The `-p` switch stands for prefix. One practical and little-documented effect of using this switch is to include the name of the function in which the code appears. This provides additional context to help Drupal users to understand the patch.

4. Post the patch to the issue queue (*Recipe 89*). Be sure to set the patch status to **patch (code needs review)**. Other users, and/or the module maintainer will test the patch to determine if the patch is *reviewed and tested by the community*—often abbreviated in online conversation as RTBC. (In other open source communities, this acronym stands for *Ready to Be Committed* but the two meanings are nearly equivalent.)

Recipe notes

- Per the maintainer's discretion, many patches will be incorporated into new module releases, while others simply remain available as options.
- More information about creating patches can be found at <http://drupal.org/patch/create>
- Windows users may use the diff utilities in the cygwin package (<http://www.cygwin.com/>) or in stand-alone utilities found at <http://unxutils.sourceforge.net/>. Windows users need to ensure that the patch file ends with a Unix line ending. Otherwise, patches generated in windows are likely to cause errors for people who attempt to apply them (test your windows patches to make sure they work).

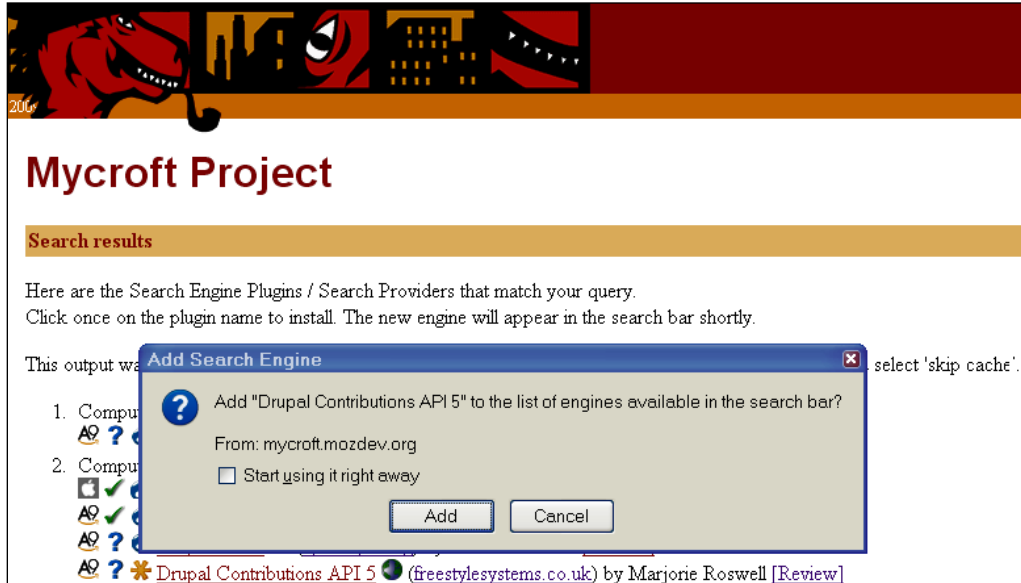
Recipe 92: Installing a Search plugin for the Contributions API

[**Ingredients**
Web Browser]

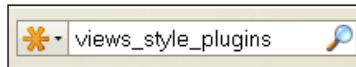
This recipe enables search of selected Drupal web sites from the web browser search form. It's a quick alternative to maintaining bookmarks to these sites. In our example we install a search plugin for the Contributed module API website at <http://api.freestylesystems.co.uk/api/5>.

The search results include files and functions found in more than two-hundred Drupal 5 modules including Views and the Views Bonus Pack. The same technique can also be used to search api.drupal.org.

1. Visit <http://mycroft.mozdev.org/search-engines.html?name=drupal>.
This performs a search of browser search plugins containing the word **drupal**.
2. Click on the Drupal Contributions API 5 link and select the **Add** button.



3. Go to the search bar in the upper-right hand corner of the browser, click on the down arrow, and select **Drupal Contributions API 5**.
4. Enter a function name, or part of a Drupal module function name, in the search box, such as **views_style_plugins**.



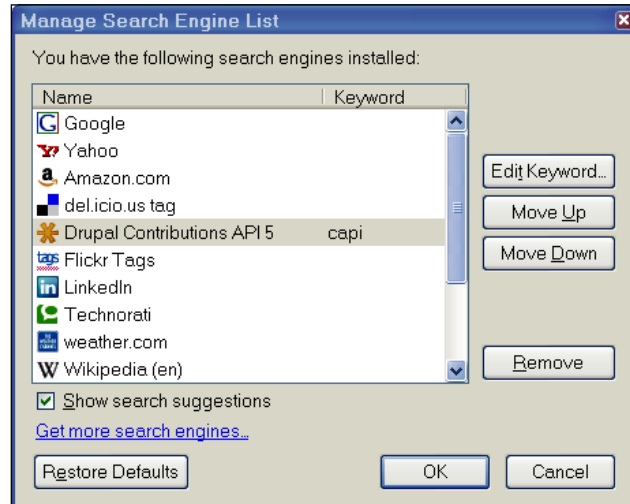
This pulls up the search results at the following URL:

http://api.freestylesystems.co.uk/api/search/5/views_style_plugins&sourceid=Mozilla-search

5. Be sure to remember to set the default search back to your favorite general search engine.

Recipe notes

- An even quicker way of using the search plugin (in lieu of changing the default search engine for each use) is to create a plugin keyword. In Firefox, click on the down arrow next to the search form, choose **Manage Search Engines...** and select **Drupal Contributions API 5**, and click **Edit Keyword**. For the contributions API search, we've entered the keyword **capi**.



Now you may enter a search phrase such as **capi views_style_plugins** or **capi views_tables** in the Address Bar. (Note the difference between the Address Bar and Search Bar, below.)



- OpenSearch plugins work in Firefox, Chrome, and Internet Explorer (Internet Explorer calls them Search Providers). If you are a Chrome browser user, click on **Options | Basics | Manage** to move the Search plugin into the Default list.
- The Mycroft site offers additional Drupal search plugins both for the standard Drupal Core API site, and for searching Drupal modules.
- Additional search plugins are available at <https://addons.mozilla.org/en-US/firefox/browse/type:4/cat:all?sort=name>.
- If you become a fan of using OpenSearch plugins, try the **Add to Search Bar** add-on for Firefox, at <https://addons.mozilla.org/en-US/firefox/addon/3682>.

Recipe 93: Using IRC—real-time technical support

[Ingredients]
Web Browser, or IRC Client

IRC is a powerful resource for Drupal users. The letters stands for Internet Relay Chat, but almost no one says those words in conversation. Most prefer to say *IRC* or *Chat*. The word **chatroom** is generally not used in the Drupal community. Each conversation area is called a **channel**.

The quickest and easiest way to participate in the IRC Drupal support channel is to follow these steps:

1. Visit <http://www.drupalchat.net>, and enter a nickname. Most people use their `drupal.org` account name (this is not required, but it does help people better know with whom they are speaking).
2. Channels load in alphabetical order. Press the **>** symbol until you see the **#drupal-support** tab (it may be a bit hard to see as it's on a dark background).

3. Click on the **#drupal-support** tab.
4. Place the cursor in the non-descript grey edit box below the conversation. Type and press enter to post a question or comment.

Recipe notes

- Be sure to read the [drupal.org](http://drupal.org/handbook/page-on-chat) handbook page on chat, <http://drupal.org/irc>.
- The most common newbie mistake is to post questions about the Drupal administrative interface on the **#drupal channel**. The handbook page clearly notes that, "Help for anything you can do with Drupal inside the web-based admin screens is best found" in **#drupal-support**.
- If you desire a more customizable experience than `DrupalChat.net`, there are many IRC clients available, including both stand-alone and browser-based clients. The ChatZilla firefox add-on is a helpful resource. A helpful ChatZilla FAQ is at <http://chatzilla.hacksrus.com/faq/>, while the add-on may be downloaded at <https://addons.mozilla.org/en-US/firefox/addon/16>. Also worth exploring in browser-based clients is `Mibbet.com`.
- Addison Berry created a terrific video guide to web-based IRC at <http://www.lullabot.com/videocast/using-irc-your-browser>.
- There are a few basic commands that you will need when using any other tool besides `DrupalChat.net`. First join the freenode network (either by clicking on a link, or setting it up as a default in your client). Use the `/join` command to load a channel as follows: `/join #drupal-support`.
- Consider registering your "nick" (Nickname) with the freenode network. Information on how to do that is available at <http://freenode.net/faq.shtml#nicksetup>.
- Several of the Drupal channels have an interesting participant, a robot ("bot") actually, named `Druplicon`. `Druplicon` seems like just another user, but this user answers certain questions instantaneously!
- There is a **#drupal-views** channel. It is not to be used for support questions, but rather for the development of Views and Panels in Drupal versions 6 and 7.
- Posts on Drupal channels are logged, and may be searched at <http://www.disobey.com/bot/log/drupal-support>, or searchirc.com.

Recipe 94: Exploring Resources

[Ingredients]

Web Browser

1. Examine the following helpful web sites and bookmark (or memorize) your favorites.

Resource Name	URL and Comments
Drupal Groups	http://groups.drupal.org Local groups connect you with other users, the best Drupal resources of all. If you search for your state or region and don't find it a group, consider starting one.
Drupal Modules	http://www.drupalmodules.com
Views Documentation	http://drupal.org/node/109604
Views 1.x Module Snippets	http://drupal.org/node/47412
Project usage overview	http://drupal.org/project/usage
Search for drupal.org users by name	http://drupal.org/search/user_search
CVS messages	http://drupal.org/project/cvs/38878 or: http://drupal.org/cvs?nid=38878
Views 1.7 Issue	http://drupal.org/node/208855
Just for Fun: Views T-shirt	http://webdesignshirts.spreadshirt.com/us/US/Shop/Article/Index/article/The-Answer-Is-Views-3560257 "I don't know the question. But the answer is Views."

Summary

The first five recipes in this chapter are related to the Drupal issue queue found at <http://drupal.org/project/issues/>, or <http://drupal.org/project/issues/<MODULENAME>>

We created a quick way to go from a lone issue number on a web page to the actual issue page. We looked at essential skills for searching and posting to the issue queue, and then covered critical Drupal skills: how to apply an existing patch, and how to create a new one. (See *Appendix G* for a categorized list of patches for Views 1.)

We looked at a nice plugin to add web site searches directly to the browser address bar. We applied this to search the Contributions API site. (It could be used just as well to `api.drupal.org`, and many other sites that contain search.)

We covered some key IRC concepts and saw a super-quick way to access the Drupal support channel. Finally, we provided a short survey of some useful web pages and sites.

Although the chapter focuses on online resources, `groups.drupal.org` will put you directly in touch with other Drupal users—close enough to shake hands and share a beer (or for the teetotalers among us, some coffee or tea). Show off your website views. Invite others to share the passions reflected in your projects. Share your knowledge. Ask questions. Above all, Enjoy.

A

Default Views in Drupal 5 Modules

We covered default views that are part of the views module in Chapter 2. We also implemented and sometimes modified default views for Audio, Calendar, Timeline, Views Bulk Operations, and Views Fast Search modules. This Appendix includes almost 175 default views.

Module Name	File Location of Default View Code	View Name
Acidfree Albums	acidfree/views_acidfree.inc	Album grid
Activity Stream	activitystream/activitystream.module	Album list delicious.com Activity Digg Activity Twitter Activity
Audio	audio/views_defaults.inc	Audio Random Audio User's Audio User's Audio Feed
Backlinks	backlinks/backlinks.module	backlinks What links here
Bibliography Module	biblio/biblio.module	Biblio
Bio	bio/bio_views.inc	Recent \$bio_name Entries Recent posts

Module Name	File Location of Default View Code	View Name
BitTorrent	bittorrent/bt_torrent.module	Torrent Files
		Popular Torrents
		Torrent Health
		Torrents
Buddylist	buddylist/buddylist_views.inc	Buddylist Content
		Buddylist
Buddylist2	buddylist2/buddy_api/buddy_api_views.inc	Buddylist
Calendar	calendar/calendar_admin.inc	Calendar
Case Tracker Work	casetracker_work/casetracker_work_views.inc	Work Node
		Work User
Competition	competition/competition.module	Open
		Latest Competitions
		All Competitions
		Entries List
		Competitions Entered by %1
Democracy forum	democracy_forum/democracy_forum_views.inc	Proposals
		New arguments
		Council Related
		Council
Devel	devel/devel_node_access.module	Nodes in the node_access table
EasyLink	easylink/easylink.module	EasyLink
erp	erp/erp_invoice/erp_invoice.module	Open invoices
		Closed invoices
		All invoices
		Open invoices to print
	erp/erp_purchase_order/erp_purchase_order.module	All purchase orders
		Closed purchase orders
		Open purchase orders
	erp/erp_item/erp_item.module	Item by category
		Item for customer
		Item nostock
		Item view

Module Name	File Location of Default View Code	View Name
	erp/erp_supplier/erp_supplier.module	suppliers
	erp/erp_job/erp_job.module	My Open Jobs
		Open Jobs
		Closed Jobs
		Submitted Jobs
		Remote Jobs
		Quote Jobs
		Onsite Jobs
		Instore Jobs
		Followups
		Calls
		My Open Jobs by type
		Appointments
	erp/erp_quote/erp_quote.module	All quotes
		Closed quotes
		Open quotes
	erp/erp_store/erp_store.module	Stores
	erp/erp_timesheet/erp_timesheet.module	Timesheet all
	erp/erp_customer/erp_customer.module	Customer
	erp/erp_price_list/erp_price_list.module	Price List
	erp/erp_cash_sale/erp_cash_sale.module	All cash sales
	erp/erp_payment/erp_payment.module	All payments
Event Views	event_views/event_views.module	Event list (drop-down date selector)
		Event list (JavaScript date selector)
		Event list (filtered by URL arguments)
Event Webform	event_webform/event_webform.module	Event webform list
FeedAPI	feedapi/feedapi_node_views/feedapi_node_views.module	Feed items
Flag	flag/flag.views.inc	My Flagname

Default Views in Drupal 5 Modules

Module Name	File Location of Default View Code	View Name
FooAggregator	fooaggregator/extra/fooaggregator_	feed
RSS aggregator	feed_page/fooaggregator_feed_page. module	
	fooaggregator/fooaggregator_views. inc	News aggregator
Forward	forward/forward.module	Most forwarded Most clickthroughs Most Active
Gallerix	gallerix/views.inc	All Albums
Housing Board	housing_board/housing_board.module	Accepted Requests/ Offers Browse Housing Offers and Requests My Offers/Requests Pending Requests/ Offers
Job Search	jobsearch/job.module	My applications Job applications Seeker applications
Kaltura	kaltura/plugins/kaltura_views/ kaltura_views.module	My Kaltura Items Most Popular Audio Most Popular Videos Most Recently Uploaded Photos Most Recently Uploaded Videos
location (API, module)	location/contrib/location_views/ location_views.module	Location table Locations User location table
Media Portfolio	media_portfolio/media_portfolio_ views.module	Show type Show awards Contest previous Contest current
MLS Module	mls/mls.module	MLS Search Result MLS MLS Listings

Module Name	File Location of Default View Code	View Name
NGP Links	ngplinks/ngplinks.module	Action Item
		Upcoming Events
		On the Issues
		Latest News
		Top Stories
		Campaign Videos
Node time tracker	node_time_tracker/node_time_tracker.module	Time_tracker
Nodeforum	nodeforum/nodeforum_views.inc	forum_replies
Node gift	nodegift/nodegift_views.module	Gifts
Node Hierarchy	nodehierarchy/nodehierarchy_views/nodehierarchy_views.module	Children teasers
		Children list
Nodequeue	nodequeue/nodequeue.module	\$queue->title
NodeReview	nodereview/nodereview_views.inc	review_list
Organic groups	og/og_views.inc	Recent posts in my groups
		Unread posts in my groups
		GHP ron
		My groups
		Groups directory
		Recent posts across whole site
		Search
		Recent comments
		Recent type term
		Galleries
OG Galleries	og_galleries/og_galleries.module	Arguments
Pro and Con arguments	procon/procon_views.inc	
Send	send/send_views.inc	History node
		Most Emailed
		Recently Emailed
Similar Nodes	similarnodes/similarnodes_example.view.inc	Similar Nodes Test

Module Name	File Location of Default View Code	View Name
SimpleFeed	simplefeed/simplefeed_item_views.inc	Feeds
		Latest feeds
		Latest feeds (Block)
Smartqueues for Organic Groups	smartqueue_og/smartqueue_og_views.inc	OG Group
Smartqueue per User	smartqueue_users/smartqueue_users_views.inc	Users user
Station	station/schedule/views_defaults.inc	Day Schedule
		Programs
		Schedule listing
		Unscheduled Programs
	station/playlist/views_defaults.inc	Playlists
	station/archive/views_defaults.inc	Archive
		Program Archive
Timeline	timeline/timeline.views.inc	Timeline of Content
		Timeline of Events
Ubercart	ubercart/uc_product/uc_product.module	Products
Update status aggregator	update_status_aggregator/update_status_aggregator.module	Module updates
Usernode	usernode/usernode.module	userlist
userreview	userreview/userreview.module	userreview
Versus	versus/versus.module	Standings by Content Type
Video	video/views_video.inc	Recent video activity
Views	views/modules/views_node.inc	frontpage
	views/modules/views_statistics.inc	Recent popular content
		All-time popular content
	views/modules/views_taxonomy.inc	Taxonomy
	views/modules/views_comment.inc	Recent Comments
	views/modules/views_user.inc	Recent posts

Module Name	File Location of Default View Code	View Name
Views Bonus Pack	views_bonus/views_bonus_panels.module	Panels by term
	views_bonus/ views_bonus_summary_combo.module	Directory
	views_bonus/views_bonus_archive.module	Archive
	views_bonus/ views_bonus_lineage_tree.module	tree
Views Bulk Operations	views_bulk_operations/ views_bulk_operations.module	Content
Views Fast Search	views_fastsearch/views_fastsearch.module	Search
Views Mail	views_mail/views_mail.module	Views Mail Opt Out
Views Multiblock	views_multiblock/views_multiblock_views.inc	Views multiblock
Webcomic	webcomic/webcomic_views.inc	Strip archive
		Latest comic
		Cast of characters
		Workflow Owner
Workflow Owner	workflow_owner/workflow_owner.module	
Yet Another Gallery Module	yagm/yagm.views.inc	Galleries
		Galleries
		Images

B

Formatters

In the Views UI, formatters display as a drop-down list of Field Options.

Formatters are discussed in Recipes 25 to 27.

Module Name	Module Page	Formatter
ASIN Field	drupal.org/project/asin	Cover and publishing info Cover image Cover image (small) Cover image (large) Publishing info only
Asset	drupal.org/project/asset	Default
Asset API	drupal.org/project/asset_api	Default Plain text Trimmed
Assetfield	drupal.org/project/assetfield	Default Imagecache Rule Names
Autonode CCK widget	drupal.org/project/autonode	Default Plain text
Body Builder	drupal.org/project/bodybuilder	Default
Brazilian IDs	drupal.org/project/brazilian_ids	Default
Content Construction Kit (CCK)	drupal.org/project/cck	Default Plain text Trimmed

Module Name	Module Page	Formatter
CCK Link to Map	drupal.org/project/ cck_link_to_map	Default
		Plain text
		Link to Map
CCK Node Reference	drupal.org/project/ cck	Title (link)
		Title (no link)
		Full node
		Teaser
CCK Number	drupal.org/project/ cck	9999
		9,999
		9,999.9
		9,999.99
		9.999
		9.999,9
		9.999,99
		9 999
		9 999, 9
		9 999, 99
		Unformatted
CCK Text	drupal.org/project/ cck	Default
		Plain text
		Trimmed
CCK User Reference	drupal.org/project/ cck	Default
		Plain text
Address field for CCK	drupal.org/project/ cck_address	Default
CCK Button	drupal.org/project/ cck_button	Button representation
CCK Download Dropdown	drupal.org/project/ cck_download_dropdown	Default
CCK Extras	drupal.org/project/ cck_extras	Selectable textarea
		Sample PHP Code
		Link redirection
		1 Pixel Out player
		Button player
		Preview image (tiny)
		Preview image (small)

Module Name	Module Page	Formatter
CCK Flashcard	drupal.org/project/cck_flashcard	Default Plain text Trimmed
CCK Formatters - Flash	drupal.org/project/cck_formatters	As Flash (SWF file)
CCK Formatters - KML	drupal.org/project/cck_formatters	As Google Map (KML file)
Fullname field for CCK	drupal.org/project/cck_fullname	Default, prefix first middle last suffix Last name only First name only Last, first middle
CCK GMap Address	drupal.org/project/cck_gmapaddress	Address Longitude Latitude Longitude and Latitude Accuracy GMap - Single GMap - All addresses in one Route Link - Single Route Link - All addresses in one GMap Route GMap User-Route
CCK IP Address	drupal.org/project/cck_ipaddr	IP address
cck_map	drupal.org/project/cck_map	Default map output
CCK Multimage	drupal.org/project/cck_multimage	CCK Multimage
CCK Node Menu	drupal.org/project/cck_nodemenu	Custom Node Menu Custom Node Menu - Single Item
CCK Redirection	drupal.org/project/cck_redirection	Default: Suppress Show as Link Show as Plain Text

Module Name	Module Page	Formatter
CCK Table Field	drupal.org/project/cck_table	Default
CCK Taxonomy Fields	drupal.org/project/cck_taxonomy	Default
CCK Taxonomy Super Select Ultra	drupal.org/project/cck_taxonomy_ssu	Default
CiviNode And Civinode CCK	drupal.org/project/civinode	Default Contact Profile Default (title) Using Profile
Colorpicker	drupal.org/project/colorpicker	Colorpicker field
Comic View	drupal.org/project/comicview	Comic Paginated Comic Paginated (Imagecache Rule Names)
CCK comment reference	drupal.org/project/commentreference	Default Plain text
Computed Field	drupal.org/project/computed_field	Raw text Plain text Markup
Content Taxonomy	drupal.org/project/content_taxonomy	As Text As Link
Craqbox	drupal.org/project/craqbox	craqbox: (Imagecache Rule Names)
CCK Currency	drupal.org/project/currency_cck	Default Plain text
Date	drupal.org/project/date	Default Long Medium Short ISO Timestamp Feed iCal As Time Ago

Module Name	Module Page	Formatter
editablefields	drupal.org/project/editablefields	Editable
Education Field	drupal.org/project/education_field	Default
Email Field	drupal.org/project/email	Default Email-Link Email-Contact Form Email-Invisimail
Embedded Media Field - Audio	drupal.org/project/emfield	Default Full Size Audio Player Preview Size Audio Player Image Thumbnail Embed Code
Embedded Media Field - Image	drupal.org/project/emfield/	Default Full Size Image Preview Size Image Image Thumbnail Embed Code
Embedded Media Field - Video	drupal.org/project/emfield	Default Full Size Video Preview Video Image Thumbnail Embed Code Thickbox: Image Thumbnail -> Full Size Video
Employment Field	drupal.org/project/employment_field	Default
Evoca Media Services - EMS	No project page, but available via CVS at contributions/modules/evoca_ms	Default player Single file player
Explainfield CCK widget	drupal.org/project/explainfield	Default Replace with explanation Value only Explanation only

Module Name	Module Page	Formatter
Eye-drop Editor (Deprecated)	drupal.org/project/eyedrop	Default Eyedrop Fieldnames
Field Reference	drupal.org/project/fieldreference	Default Full Teaser
FileField	drupal.org/project/filefield	Default
Fivestar	drupal.org/project/fivestar	As Stars Rating (i.e. 4.2/5) Percentage (i.e. 92)
Flickr	drupal.org/project/flickr	photo_id set_id
Geonames CCK	drupal.org/project/geonames_cck	Default No geotag Hidden geo code With disambiguation link
GMap Addons	drupal.org/project/gmap_addons	Default Map GSIV
Giant Scalable Image Viewer	drupal.org/node/187640	GSIV: Imagecache Rule Names
Guitar	drupal.org/project/guitar	Default
ImageCache	drupal.org/project/imagecache	Default Preset Name Preset Name as link to node Preset Name as link to image Path to Preset Name derivative URL to Preset Name derivative

Module Name	Module Page	Formatter
ImageField	drupal.org/project/imagefield	Default link to node link to image path to image url to image
Inventory field	drupal.org/project/inventory	Default Skip Empty Fields
jQuery Lightbox	drupal.org/project/jlightbox	jLightbox: Imagecache Rule name gallery jLightbox: Imagecache Rule name
jQuery Gallery View for CCK multiple imagefield	drupal.org/project/jqgalview	jqGalView: Preset name
JQZoom	drupal.org/project/jqzoom	JQZoom Imagecache Rule name
JSNodeLoad	drupal.org/project/jsnodeload	Reference + text Text only Reference only
Lightbox2	drupal.org/project/lightbox2	Lightbox2 iframe: original->node page Lightbox2: original ->View Rule Lightbox2: View Rule ->original Lightbox2 slideshow: original->View Rule Lightbox2 slideshow: View Rule->original Lightbox2: View Rule ->Lightbox Rule Lightbox2 slideshow: View Rule->Lightbox Rule

Module Name	Module Page	Formatter
		Lightbox2 iframe:View Rule->node page Lightbox2: Image Thumbnail -> Original Lightbox2 slideshow: Image Thumbnail -> Original Lightbox2 iframe: Image Thumbnail -> Content Lightbox2: Image Thumbnail -> Full Size Video Lightbox2 iframe
Link	drupal.org/project/ link	Title, as link (default) URL, as link URL, plain text Short, as link with title "Link" Label, as link with label as title Separate title and URL
Location (API, module)	drupal.org/project/ location	Default
Matrix Field	drupal.org/project/ matrix	Table View
Media Field - Audiofield	drupal.org/project/ mediafield	Default
Media Field - Videofield	drupal.org/project/ mediafield	Default
Media Field - Display	drupal.org/project/ mediafield_display	1 Pixel Out player 1 Pixel Out player plus download link Button player Button player plus download link

Module Name	Module Page	Formatter
Money CCK field	drupal.org/project/money	3-letter code after amount 3-letter code before amount Full name after amount Full name before amount Symbol after amount Symbol before amount
Multireference CCK field	drupal.org/project/multireference	list table
Namefield	drupal.org/project/namefield	Default
Office Hours	drupal.org/project/office_hours	Default
OpenPackage Video	drupal.org/project/op_video	Inline player Preview image - link to node Preview image - play in layer
Phone (CCK)	drupal.org/project/phone	Default
Pollfield	drupal.org/project/pollfield	Default All choices All results Voting widget Question Runtime Active Choice item (Views only) Vote item (Views only)
Postal	drupal.org/project/postal	Default

Module Name	Module Page	Formatter
Preset Widget Options	drupal.org/project/preset_widget_options	Creative Commons license (when installed with http://drupal.org/project/cc_widget)
Private Number	drupal.org/project/private_number	Default Plain text
Role Reference	drupal.org/project/rolereference	Default Plain text
See Map	drupal.org/project/see_map	Default
Shadowbox	drupal.org/project/shadowbox	shadowbox: original to View Rule shadowbox: View Rule to original shadowbox gallery: original to View Rule shadowbox gallery: View Rule to original shadowbox: View Rule to Shadowbox Rule shadowbox gallery: View Rule to Shadowbox Rule
SMS Framework	drupal.org/project/smsframework	SMS Link
State Reference	drupal.org/project/statereference	Default Plain text
Text Field Tags	drupal.org/project/text_field_tags	Heading 1 Heading 2 Heading 3 Heading 4 Bold Italic Underline Block quote (indent) As link
Thickbox	drupal.org/project/thickbox	Thickbox: Preset Name

Module Name	Module Page	Formatter
Timer	drupal.org/project/timer	Default
Trackfield	drupal.org/project/trackfield	unformatted
Trackfield - Graph	drupal.org/project/trackfield	Format Sizes
Trackfield - Map	drupal.org/project/trackfield	Map Format Sizes
Trackfield - Stats	drupal.org/project/trackfield	Default Metric - Auto 9999m 9.99Km 9.9% hh:mm:ss 99h99m99s
Ubercart Add to Cart CCK Field	drupal.org/project/uc_cart_widget	Add to Cart (w/attributes) Add to Cart (no attributes)
VCalfield	drupal.org/project/vcalfield	VCalendar
Video Upload	drupal.org/project/video_upload	Default Thumbnail Image Thumbnail Image as link Small Video
Viewfield	drupal.org/project/viewfield	Use view "Page" settings Use view "Block" settings Count of items in view Style plugins
zipcode (CCK)	drupal.org/project/zipcode	Default
Zoomify	drupal.org/project/zoomify	Zoomify Zoomify: Imagecache Rule Names

C

Style Plugins

Style plugins populate the **View Type** drop-down in the **Views UI**. They offer sophisticated formatting options for nodes in a page and a block

Full Name	URL	Style Plugin
Action Views	drupal.org/project/action_view	Action View
Active Template	drupal.org/project/active_template	Template (Full Nodes) Template (Node Teasers) Template (Selected Fields)
Activity Stream	drupal.org/project/activitystream	Activity Stream
Ad Views (Module author in search of a maintainer)	drupal.org/project/ad_views	Advertisement
AJAX Views	drupal.org/project/ajax_views	AJAX Paging Block View
Asset API	drupal.org/project/asset_api	AAPF XSPF: XSPF Playlist Views RSS: iTunes RSS feed
Bibliography module	drupal.org/project/biblio	Biblio: Classic Biblio: APA Biblio: CSE Biblio: IEEE
Calendar	drupal.org/project/calendar	Calendar
Carousel	drupal.org/project/carousel	Carousel View
Conference Organizing	drupal.org/project/conference_organizing	COD: Schedule Table COD: Schedule List

Style Plugins

Full Name	URL	Style Plugin
Date	drupal.org/project/date	Date: Date Browser
Dynamic Views	drupal.org/project/dynamic_views	Dynamic Full Node Dynamic Filter: Teaser List Dynamic Filter: Full Nodes Dynamic Filter: List Dynamic Filter: Table
e-Commerce	drupal.org/project/ecommerce	Product: List of products
editablefields	drupal.org/project/editablefields	Editablefields - table Editablefields - list Editablefields - table - no form Editablefields - list - no form
Editview	drupal.org/project/editview	Editview - compressed Editview - table
jCarousel Lite	drupal.org/project/jcarousellite	jCarousel Lite View
Drupal for facebook	drupal.org/project/fb	facebook Discussion facebook Teaser List
FLV Media Player	drupal.org/project/flvmediaplayer	FLV Media Player recommended content (Views integration removed in November, 2008. Not necessary for the JW 4.x player.)
FooAggregator RSS aggregator	drupal.org/project/fooaggregator	Fooaggregator Admin
Frontpage Slideshow	drupal.org/project/fpss	Frontpage Slideshow
FusionCharts	drupal.org/project/fusioncharts	FusionChart view
Gmap Module	drupal.org/project/gmap	Gmap View
Google Maps Tools	Drupal.org/project/gmaps	Google Maps View
gProximity	drupal.org/project/gproximity	gProximity: Google Map Locations
Image	drupal.org/project/image	Image: Gallery
Imageflow	drupal.org/project/imageflow	Imageflow View

Full Name	URL	Style Plugin
KML module	drupal.org/project/kml	KML feed
Views: Many Node Export	drupal.org/project/many_node_export_views	Views Export: CSV file
Node Cloud	drupal.org/project/nodecloud	Node Cloud
Node comments	drupal.org/project/nodecomment	Node comment: Threaded comments
Node Reference Views Select	drupal.org/project/nodereference_views_select	Sortable teaser list for Node Reference
Organic Group Galleries	drupal.org/project/og_galleries	OG Galleries
OpenPackage Video	drupal.org/project/op_video	Video jCarousel
oyoaha Liquid Layout	drupal.org/project/oaliquid	Liquid: List of Fields Liquid: List of Nodes
Panel Style	drupal.org/project/panel_style	Panel Titles
Partial	drupal.org/project/partial	Partial: View Styles
Project issue tracking	drupal.org/project/project_issue	Project issues table
SlideShowPro	drupal.org/project/slideshowpro	SlideShowPro: XML feed
Timeline	drupal.org/project/timeline	Timeline Vertical Timeline Horizontal
Views	drupal.org/project/views	List View Table View Teaser List Full Nodes Views RSS: RSS feed
Views Bonus Pack	drupal.org/project/views_bonus	Panels: Teasers, 1 top + 2 columns Panels: Teasers, 2 columns Panels: Teasers, 1 top + 3 columns

Style Plugins

Full Name	URL	Style Plugin
		Panels: Teasers, 3 columns
		Panels: By Term, 3 columns
		Bonus: Grid View
		Bonus: Summary + full view
		Lineage: Nested taxonomy summary
		Audio: Playlist
		Bonus: Plain View
		Views Bonus: CSV file
		Views Bonus: DOC file
		Views Bonus: TXT file
		Bonus: Sparkline View
Views Bulk Operations (VBO)	drupal.org/project/views_bulk_operations	Bulk Operations View
Views Calc	drupal.org/project/views_calc	Calc Table View
Views Datasource	drupal.org/project/views_datasource	Views JSON: Simple JSON data document Views JSON: MIT Simile/Exhibit JSON data document Views RDF: FOAF RDF data document Views XHTML: hCard XHTML data document Views XML: Raw XML data document Views XML: OPML XML data document
Views carousel	drupal.org/project/viewscarousel	Carousel (See also Carousel.)
Views Fast Search	drupal.org/project/views_fastsearch	Search Results
Views 'Group-By' Pack	drupal.org/project/views_groupby_pack	Grouped By Month, Full Nodes Grouped By Month, Teasers Grouped By Taxonomy Term, Full Nodes Grouped By Taxonomy Term, Teasers

Full Name	URL	Style Plugin
Views JSON	http://drupal.org/node/131655/cvs-instructions/DRUPAL-5	Views JSON: JSON Response
Views Mail	drupal.org/project/views_mail	Views mail
Views_Multiblock	drupal.org/project/views_multiblock	Views_Multiblock
Views Node Feed	drupal.org/project/views_node_feed	Views Node Feed: Node feed
View of Views	drupal.org/project/view_of_views	View_Of_Views - table
Views Podcast	drupal.org/project/views_podcast	Views Podcast: Podcast feed
Views Popup	drupal.org/project/views_popup	List View as Popup, Show 1 field List View as Popup, Show 2 fields List View as Popup, Show 3 fields
Views Rotator	drupal.org/project/views_rotator	Views Rotator - Items Views Rotator - Nodes Views Rotator - Teasers
Views Slideshow	drupal.org/project/views_slideshow	Slideshow List Slideshow Teasers Slideshow Full Nodes
Views Style Plugin Options	drupal.org/project/vspo	Nice Title
Views Tabs	drupal.org/project/views_tabs	Tabs - Fields Tabs - Teaser List Tabs - Full Nodes
Views Tagadelic	drupal.org/project/tagadelic_views	Tagadelic

Style Plugins

Full Name	URL	Style Plugin
Views Ticker	drupal.org/project/ views_ticker	Ticker: Fade Ticker: BBC Style Ticker: Scroller (Horizontal) Ticker: Scroller (Vertical)
XML Sitemap	drupal.org/project/ xmlsitemap	XML Sitemap: Sitemap XML Sitemap: News
XSPF Playlist	drupal.org/project/ xspf_playlist	XSPF Playlist node feed
Yet Another Gallery Module	drupal.org/project/yagm	Image Gallery

D

Views Hooks for Coders

The following hooks are frequently used by module developers.

Hooks	Description
<code>hook_views_arguments()</code>	Create Views arguments array, including name, help text, handler code, and options
<code>hook_views_default_views()</code>	Create default Views (See Chapter 2, and <i>Appendix A</i>).
<code>hook_views_style_plugins</code>	Create array of style plug-ins. These style plug-ins populate the View Type selection list in the Views UI. (See also Chapter 4, and <i>Appendix C</i> .)
<code>hook_views_tables()</code>	Create tables objects for use by Views The most common Views hook. See http://drupal.org/handbook/modules/views/api for more information. Be sure to read Parts 1 to 5.

The following hooks tend to be less well-known and not used as often:

Hooks	Description
<code>hook_views_arguments_alter()</code>	Allow modules to alter the arguments supplied others See http://drupal.org/node/105620 for more information
<code>hook_views_feed_argument()</code>	Alter Views feed argument Sample use-case at http://drupal.org/node/333039 . Also used by the Existing Nodes Filter module, at http://drupal.org/project/existingnodesfilter
<code>hook_views_post_view()</code>	Alter the output string after the view is rendered. Used by Views' own <code>views_rss</code> module, and by the Calendar module.

Hooks	Description
<code>hook_views_pre_view()</code>	Alter returned data before it is rendered. Sample use-cases in the Calendar module, and at http://www.onetomany.com/adding_node_counts_view_titles_views_1x .
<code>hook_views_pre_query()</code>	Alter or extend the view query. Used by the i18n and Drupal For Firebug modules.
<code>hook_views_query_substitution()</code>	Run-time query replacement. Used by a new module called Views Menu Nodes, http://drupal.org/project/views_menu_nodes .
<code>hook_views_tables_alter()</code>	Alter views table values. Sample use case for this little-known hook at http://drupalhigh.onsugar.com/2669697 .
<code>hook_views_tabs()</code>	Array of tabs. Used by the Calendar module
<code>hook_views_theme_wizard_types()</code>	Add new theme wizard types. Used, naturally, by the Views Theme Wizard module, but like all of these hooks, available for use by other modules, as well.

This table displays relevant hooks from other modules:

Hooks	Description
<code>hook_field_formatter()</code> (From the CCK module)	In the Views UI, formatters created by this hook display as a drop-down list of Field Options. (See <i>Appendix B</i> .)
<code>hook_views_fusion()</code> (From the Views Fusion module)	Provide information on how to join the node tables from other modules. (See <i>Recipe 55</i> .)

E

Modules Included in Recipe Ingredients

Module	URL	Recipe	Maintainer	Updated to Drupal 6?
Actions	drupal.org/ project/ actions	51	jvandyk	Built into Drupal 6
Administration menu	drupal.org/ project/admin_ menu	45, 50	sun	Yes
Admin Role	drupal.org/ project/ adminrole	7	JacobSingh	Yes
Audio	drupal.org/ project/audio	64	drewish	Yes
Automatic Nodetitles	drupal.org/ project/auto_ nodetitle	31	fago	Yes
Backup and Migrate	drupal.org/ project/ backup_migrate	44	ronan	Yes
Calendar	drupal.org/ project/ calendar	42, 43, 44	KarenS	Yes
Calendar Popup iCal		42 43		
Carousel	drupal.org/ project/ carousel	83	Omar	No

Modules Included in Recipe Ingredients

Module	URL	Recipe	Maintainer	Updated to Drupal 6?
CCK Extras	drupal.org/project/cck_extras	25, 26	eaton	No. You may add a patch at drupal.org/node/296191
Computed Field	drupal.org/project/computed_field	32, 81	Moonshine	Yes
Content Construction Kit (CCK)	drupal.org/project/cck	Throughout	yched	Yes
Node Reference		29, 30, 31, 56		
Country codes API	drupal.org/project/countries_api	61,62	mrfelton	Yes
Date Date API	drupal.org/project/date	29, 30, 33-44, 80, 81	KarenS	Yes
Devel	drupal.org/project/devel	32, 54, 72, 73	moshe weitzman	Yes
editablefields	drupal.org/project/editablefields	49	markfoodyburton	Yes
Field Copy	drupal.org/project/field_copy	24	regx	No
Flag	drupal.org/project/flag	65, 66	quicksketch	Yes
getID3()	drupal.org/project/getid3	64	Rob Loach	Yes
Google Maps Tools	drupal.org/project/gmaps	61,62	xmarket	Yes
Guitar Chords	drupal.org/project/guitar	27	flobruit	Yes
ImageAPI ImageAPI GD2	drupal.org/project/imageapi	59, 60	drewish	Yes
ImageCache	drupal.org/project/imagecache	59, 60	drewish	Yes

Module	URL	Recipe	Maintainer	Updated to Drupal 6?
ImageField	drupal.org/ project/ imagefield	31, 59, 60	quicksketch	Yes
jCarousel	drupal.org/ project/ jcarousel	83	Wim Leers	Yes
jQuery Update	drupal.org/ project/ jquery_update	60, 61, 83	jjeff	Yes
Javascript Tools	drupal.org/ project/ jstools	84	nedjo	Yes
Lightbox2	drupal.org/ project/ lightbox2	60	stella	Yes
Link	drupal.org/ project/link	25, 26, 29, 30	jcfiala	Yes
Masquerade	drupal.org/ project/ masquerade	47	Gurpartap Singh	Yes
ModuleInfo	drupal.org/ project/ moduleinfo	46	jabapyth	Yes
Node Go To module	drupal.org/ project/ nodegoto	38	introfini	Not yet, but watch D6 patch, http:// drupal.org/ node/290383
Panels	drupal.org/ project/panels	70, 71, 72, 73	merlinofchaos	Yes, beta release
Pathauto	drupal.org/ project/ pathauto	64	greggles	Yes
PhoneBlogz	drupal.org/ project/ phoneblogz	67	mattb	No

Modules Included in Recipe Ingredients

Module	URL	Recipe	Maintainer	Updated to Drupal 6?
Poormanscron	drupal.org/ project/ poormanscron	53	Uwe Hermann	Yes
Quick Tabs	drupal.org/ project/ quicktabs	85	katbailey	Yes
Read More link	drupal.org/ project/ed_ readmore	13	Todd Nienkerk	Yes
Regenerate	drupal.org/ node/195013# comment- 1168380	32		Yes, in development on same page
sIFR	drupal.org/ project/sifr	32	sun	No, but newer drupal.org/ project/render module will be ported
SimpleFeed	drupal.org/ project/ simplefeed	70, 71, 72, 73	m3avrck	Yes, in development
Statistics	Drupal Core (drupal.org)	14, 15, 22		Yes
Taxonomy Manager	drupal.org/ project/ taxonomy_ manager	16	mh86	Yes
Timeline	drupal.org/ project/ timeline	40	jmiccolis	Yes
Token	drupal.org/ project/token	28, 31, 64	eaton	Yes
Ubercart	drupal.org/ project/ ubercart	28	Andy_Lowe	Yes
Viewfield	drupal.org/ project/ viewfield	28	jerdavis	Yes

Module	URL	Recipe	Maintainer	Updated to Drupal 6?
Views	drupal.org/ project/views	Throughout	merlinofchaos (with much Views 1 maintenance by Sun)	Yes
Views Theme Wizard		79, 81	merlinofchaos	No
Views Bonus Pack	drupal.org/ project/views_ bonus	58, 59, 60	dmitrig01	Yes
Bonus Grid View		59, 60		
Views Bonus Export		58		
Views Bulk Operations (VBO)	drupal.org/ project/ views_bulk_ operations	50, 51	kratib	Yes
Views Calc	drupal.org/ project/views_ calc	63	KarenS	Yes
Views Checkboxes	drupal.org/ project/views_ checkboxes	67	nschelly	
Views Custom Field	drupal.org/ project/views_ customfield	52, 62, 66	casey	Yes
Views Date Range Filter Module	drupal.org/ project/ daterange	36	douggreen	No
Views Fast Search	drupal.org/ project/views_ fastsearch	56	douggreen	No
Views Fusion	drupal.org/ project/views_ fusion	55	fago	No. Relationships are built into Views 2
Views Popup	drupal.org/ project/views_ popup	41	bwong	Yes

Module	URL	Recipe	Maintainer	Updated to Drupal 6?
Views RSS	drupal.org/project/views	12	merlinofchaos	Yes
Views UI permissions	drupal.org/project/views_ui_perm	47	Abandoned Modules (You may adopt it See drupal.org/node/251466 .)	No
Views UI Sort	drupal.org/project/views_ui_sort	48	regx	No

F

Additional Resources and Modules Mentioned in Recipes

Resource or Module	URL	Recipe	Maintainer
Administration Menu Dropdown	drupal.org/project/admin_menu_dropdown	45	Shawn Conn
Chrome browser	http://google.com/chrome	54	
ColorZilla	http://addons.mozilla.org/en-US/firefox/addon/271	42, 74	
Couloir Slideshow	drupal.org/project/couloir_slideshow	83	ComputerWolf
Drupal For Firebug	drupal.org/project/drupalforfirebug	54	populist
Dynamic Rendering	drupal.org/project/render	32	sun
The Everything Book	http://visibone.com		
Excerpt	drupal.org/project/excerpt	13	hayesr
Fading Slideshow	drupal.org/project/fade_slideshow	83	der
Favicon Generator	http://antifavicon.com/	45	
Featured Content Slider	drupal.org/project/content_slider	83	ebizondrupalservices

Additional Resources and Modules Mentioned in Recipes

Resource or Module	URL	Recipe	Maintainer
FeedBurner.com	http://feedburner.com	64	
Firebug	http://getfirebug.com	8, 9, 40, 69	
Firefox	http://mozilla.com/firefox	8, 9, 42	
Google Analytics	http://google.com/analytics	14	
Instant SQL Formatter	http://wangz.net/gsqlparser/sqlpp/sqlformat.htm	54	
Jquery Slideshow	drupal.org/project/jquery_slideshow	83	spydor
Kaltura	drupal.org/project/kaltura	57	gonen.kaltura
Views: Many Node Export	drupal.org/project/many_node_export_views	58	mfer
Nodequeue	drupal.org/project/nodequeue	12	ezra-g
Organic groups	drupal.org/project/og	67	moshe weitzman
Pager Preference	drupal.org/project/pager_preference	75	v1nce
Paranoia	drupal.org/project/paranoia	71	kill@www.drop.org
Path	Drupal Core	64	
Porter-Stemmer	drupal.org/project/porterstemmer	56	greggles
PSPad	http://www.pspad.com/en/download.php	54	
SIMILE Timeline	http://simile.mit.edu/timeline/examples/ http://simile-widgets.org/timeline/	42	
Site notes	drupal.org/project/sitenotes	40	NancyDru

Resource or Module	URL	Recipe	Maintainer
Slideshow	drupal.org/project/slideshow	83	kkaefer
SlideShowPro	drupal.org/project/slideshowpro	83	alex_b
Taxonomy	Drupal Core	16,51	
Taxonomy Access Control Lite	drupal.org/project/tac_lite	40	Dave Cohen
Taxonomy Hide	drupal.org/project/taxonomy_hide	16	brmassa
Taxonomy Manager	drupal.org/project/taxonomy_manager	16	mh86
Taxonomy Redirect	drupal.org/project/taxonomy_redirect	16	Agileware
Timeline Widget	http://miccolis.net/random/timeline-1-2.tar.gz	42	jmccolis
Tracker	Drupal Core	21	
View Alias	drupal.org/project/view_alias	64	emackn
Views: Many Node Export	drupal.org/project/many_node_export_views	58	mfer
Views Rotator	drupal.org/project/views_rotator	83	mfer
Views Slideshow	drupal.org/project/views_slideshow	83	aaron
View Source Chart	http://jennifermadden.com/scripts/ViewRenderedSource.html	69	Jennifer Madden
Views Tabs	drupal.org/project/views_tabs	84	nedjo
Views Taxargs	drupal.org/project/views_taxargs	16	linuxbox
View Title Count	view_title_count.zip , at http://michaelanello.com/adding_node_counts_view_titles_views_1x	78	Micheal Anello

G

Selected Noteworthy Patches to Views

To apply a patch, go to the command line in the module directory (usually `sites/all/modules/views` or one of its subdirectories) and type:

```
patch < filename.patch
```

These patches are not guaranteed. Be sure to read the issue posts, test the patches in a development environment, and report your results back to the issue queue.

A complete list of Views 1 patches is available at:

[http://drupal.org/project/issues/search/views?status\[\]=13&status\[\]=8&status\[\]=14&status\[\]=15&version\[\]=5.x](http://drupal.org/project/issues/search/views?status[]=13&status[]=8&status[]=14&status[]=15&version[]=5.x)

Patch name	URL
Arguments	
'node: id' argument handling fails on str type values	drupal.org/node/263855
Can't have taxonomy argument and filter	drupal.org/node/277919
Arguments Type Taxonomy Name of Term	drupal.org/node/267435
Add a 'Node: Posted Day' argument	drupal.org/node/150808
Add argument handler for "Front page promoted status"	drupal.org/node/182128
Argument in the beginning of the URL is processed incorrectly	drupal.org/node/180776
Author Role Argument	drupal.org/node/159850
Comparison with NULL fails	drupal.org/node/279478
Node ID argument summary not working	drupal.org/node/238610
Support Day of Month argument	drupal.org/node/199366
using two taxonomy term names as arguments in views	drupal.org/node/77543

Patch name	URL
views_handler_arg_nodetype() escapes its argument twice	drupal.org/node/293068
well defined paths should override argument based paths	drupal.org/node/170178
Breadcrumbs	
Breadcrumb - not replaced % mark in the title of views.	drupal.org/node/201704
Malformed menu/breadcrumb when view contains Node: Body field	drupal.org/node/176360
Tokens not substituted in breadcrumb	drupal.org/node/183191
Comments	
Correct 'Comment: Link' for paged comment lists	drupal.org/node/236264
Comments Distinct	drupal.org/node/219499
Missing plural in new comments mark	drupal.org/node/206934
views_comments: Comments with no subjects have no clickable link text	drupal.org/node/282993
Export	
\$view->menu_tab_default_parent_type not getting exported	drupal.org/node/171067
Bulk export for Drupal 5/Views 1	drupal.org/node/402636
Exported views use of "array" (not up to coding standards)	drupal.org/node/174044
Fields	
Add field to 'contact' node authors	drupal.org/node/154865
New field for node published status	drupal.org/node/245482
Node: Body outputs entire full node body instead of Body field	drupal.org/node/160641
Truncate node teaser/body field to custom length	drupal.org/node/177082
UID field for views_user.inc	drupal.org/node/339885
Expose node author's email address as a field	drupal.org/node/177494
Port usernode user table definitions to views_user.inc	drupal.org/node/143356
Path (alias) integration	drupal.org/node/257046

Patch name	URL
Filters	
"None of" (NOR) filter incorrect on vocabulary name	drupal.org/node/225899
Can't have taxonomy argument and filter	drupal.org/node/277919
Add timestamp support to timestamp filters	drupal.org/node/333123
Adding a range filter operator	drupal.org/node/151989
Exposed filters: URL processing on embeds	drupal.org/node/156130
Filter 'Node: Author Name (text field)'	drupal.org/node/199162
Issue with "Search: Index" exposed filter	drupal.org/node/365656
Node: Created Month' and 'Node: Created Year' filters	drupal.org/node/203407
Adding same filter twice doesn't work	drupal.org/node/250769
clean up filter code to work with alias prefixes	drupal.org/node/150094
Filter nodes by whether author account is active/ blocked	drupal.org/node/277963
Node body length filter	drupal.org/node/188376
Path (alias) integration	drupal.org/node/257046
Port usernode user table definitions to views_user.inc	drupal.org/node/143356
views filter Author is Anonymous is broken	drupal.org/node/226570
Views filter to accept timestamps	drupal.org/node/355420
Programmatic Filters Failing	drupal.org/node/150517
RSS	
extra data necessary for extra_RSS_fields	drupal.org/node/341240
messages generated during rss feed	drupal.org/node/175436
Option to anonymize RSS feeds in Views	drupal.org/node/335934
RSS feeds from views bypass the page cache	drupal.org/node/231424
RSS link on summary view using '*' instead of custom Wildcard	drupal.org/node/102746
Views RSS: view hook results not captured	drupal.org/node/196933
Views_rss doesn't apply input filters to fulltext output	drupal.org/node/147821
Sort	
Add node handler to sort by promoted to front page	drupal.org/node/262916
Expose node author's email address as a field	drupal.org/node/177494
How do i add the nid from term_node as a sort criteria	drupal.org/node/433396
Path (alias) integration	drupal.org/node/257046

Patch name	URL
Port usernode user table definitions to views_user.inc	drupal.org/node/143356
tablesort_sql in _views_query->query() escapes characters in output of sort_handler functions causing SQL errors	drupal.org/node/231796
Views_search: Sort by relevance	drupal.org/node/365566
Views sort parameters override table sort.	drupal.org/node/259083
SQL	
"Is Not Equal To" for LIKE comparisons	drupal.org/node/169709
Add HAVING support	drupal.org/node/170245
Cached queries not available in hook_views_pre_query	drupal.org/node/153924
query limit \$start, \$count	drupal.org/node/125015
support subqueries	drupal.org/node/143888
tablesort_sql in _views_query->query() escapes characters in output of sort_handler functions causing errors	drupal.org/node/231796
Taxonomy	
Arguments Type Taxonomy Name of Term	drupal.org/node/267435
Can't have taxonomy argument and filter	drupal.org/node/277919
using two taxonomy term names as arguments in views	drupal.org/node/77543
Missing taxonomy terms in filters (Covered in Recipe 17)	drupal.org/node/199675
Taxonomy Term distinct	drupal.org/node/195790
Taxonomy term id argument breaks without \$depth option	drupal.org/node/221000
Taxonomybased view SQL woes	drupal.org/node/298542
Views 1.7 pre-req #2: Write new vocabulary fetcher	drupal.org/node/272289
Theme	
enable stripes in the views theme wizard	drupal.org/node/268528
Provide \$info to views theme function	drupal.org/node/160381
This patch adds a theme function for 'number of new comments'	drupal.org/node/210933

Patch name	URL
Other Views Patches	
'more' link appears in block when block is exactly full	drupal.org/node/282508
Add creating module to list of default views	drupal.org/node/237272
Alphabetic summary listing of node titles causes SQL error	drupal.org/node/171565
Ampersand escaped twice in links to files	drupal.org/node/212853
API cannot parse comments	drupal.org/node/198293
Avoid prefixing temporary tables	drupal.org/node/129838
Checkbox data does not display properly on Windows w/Garland	drupal.org/node/167439
Extend the '\$node' token to accept multiple node types	drupal.org/node/211093
Little UI typo fix	drupal.org/node/354747
node_invoke_nodeapi is called with incorrect \$a4 (page)	drupal.org/node/385742
Node Status Value Should Not Be Hardcoded	drupal.org/node/382872
Search index with no results gives no empty text	drupal.org/node/330481
Small table accessibility improvement	drupal.org/node/183592
Strange behavior w/ blog views, pages titled "My blog"	drupal.org/node/128520
static cache views for views that can't be cached	drupal.org/node/236809
Translate views with i18n	drupal.org/node/245124
Use block title (instead of view's name) in "Configure blocks"	drupal.org/node/257575
view_node.inc handlers not table alias safe	drupal.org/node/142714
Views_search: php-error when nothing was found	drupal.org/node/365013
Views cache too large	drupal.org/node/218187
Views handler for node 'edit' and 'delete' links' node access is faulty	drupal.org/node/325357
Views menu items cannot be translated	drupal.org/node/234153
Wrong feed url when using ags handling code!	drupal.org/node/225416
Other	
Views fusion support for nodereference	drupal.org/node/123482
Color Coding based on taxonomy or content type	drupal.org/node/121298
Configurable icons for timeline events (Timeline module)	drupal.org/node/104890

Index

Symbols

#drupal-views channel 331
*.tpl.php files 139
-p switch 327
-u switch 327

A

Acidfree Albums, default views 335
actions module 363
action views, style plugins 355
active template, style plugins 355
activity stream, default views 335
activity stream, style plugins 355
Address field for CCK, formatters 344
administration menu dropdown
 module 177, 369
administration menu module 363
 about 175
 features 176, 177
 recipe notes 177
 URL 175
admin role module
 about 363
 adding 31
 configuring 32
 installing 30
 user, creating 32
AJAX views, style plugins 355
archived listerv, URL 156
arguments, patch name 373
ASIN Field, formatters 343
Asset, formatters 343
Asset API, formatters 343
asset API, style plugins 355
Assetfield, formatters 343

audio, default views 335
audio module 363
 content, adding 243, 244
 recipe notes 247
 setting up 242
 URL 242
 view, setting up 245, 246
automatic nodetitles module 363
Autonode CCK widget, formatters 343

B

Backlinks, default views 335
backup and migrate module 363
 about 363
 URL 168
bibliography module, default views 335
bibliography module, style plugins 355
Bio, default views 335
BitTorrent, default views 336
blocks 27
blogroll, creating
 blogroll block, placing 107
 link field, modifying 106
 recipe notes 107
 view creating, website_snapshots
 based 107
Body Builder, formatters 343
body classes
 about 304
 recipe notes 305
Bonus Grid View 367
Brazilian IDs, formatters 343
breadcrumbs, patch name 374
Buddylist, default views 336
Buddylist2, default views 336

C

- calendar, default views 336
- calendar, style plugins 355
- calendar 2.x module, URL 168
- calendar module
 - about 363
 - calendar blocks, installing 162, 163
 - configuring 160
 - installing 160
 - recipe notes 163, 164
 - URL 159, 163
 - view setup 160, 161
- calendar popup module 363
- carousel, style plugins 355
- carousel module
 - about 363
 - URL 306
- Case Tracker Work, default views 336
- CCK-related modules list, URL 95
- cck_map, formatters 345
- CCK Button, formatters 344
- CCK comment reference, formatters 346
- CCK Currency, formatters 346
- CCK Download Dropdown, formatters 344
- CCK Extras, formatters 344
- CCK extras module 364
- CCK Flashcard, formatters 345
- CCK Formatters - Flash, formatters 345
- CCK Formatters - KML, formatters 345
- CCK GMap Address, formatters 345
- CCK IP Address, formatters 345
- CCK Link to Map, formatters 344
- CCK module, URL 95, 140
- CCK Multimage, formatters 345
- CCK Node Menu, formatters 345
- CCK Node Reference, formatters 344
- CCK Redirection, formatters 345
- CCK Table Field, formatters 346
- CCK Taxonomy Fields, formatters 346
- CCK Taxonomy Super Select Ultra, formatters 346
- CCK Text, formatters 344
- CCK User Reference, formatters 344
- channel 330
- ChatZilla FAQ, URL 331
- chrome browser 369
- Chrome Browser, URL 201
- CiviNode And Civinode CCK, formatters 346
- Clear Views Cache 52
- Colorpicker, formatters 346
- ColorZilla
 - about 369
 - URL 164
- Comic View, formatters 346
- command line tips 11
 - !\$ 12
 - cd 12
 - Ctrl+K 12
 - Ctrl+R 12
 - Ctrl+U 11
 - less !\$ 12
 - man less 12
 - pwd 12
- comments, patch name 374
- comments_recent, default views
 - block, configuring 45-47
 - date field, configuring 47
 - existing views 44
 - Recent comments blocks 44
 - Recent comments blocks, by comment module 44
 - Recent comments blocks, by views module 44
 - view, overriding 45
- competition, default views 336
- computed Field, formatters 346
- computed field module 364
 - about 302, 303
 - recipe notes 303
 - URL 302
- conference organizing, style plugins 355
- Content Construction Kit. *See* CCK module
- Content Construction Kit (CCK), formatters 343
- CCK module 364
- content module, URL 218
- Content Taxonomy, formatters 346
- content type
 - relating, with reverse node reference 119
- control icon configurations, URL 230
- Couloir Slideshow 369

country codes API module

- about 364
- URL 225

craigslist, URL 265

Craqbox, formatters 346

cron setting up

- cPanel used 197-199
- on Linux box 199
- Poormanscron module used 200

custom field module

- about 251, 252
- URL 250

CVS messages, URL 332

D

Date, formatters 346

date, style plugins 356

date 2.x module, URL 168

date and calendar module, upgrading

- 1.x and 2.x versions differences, URL 168
- about 170-173
- database backing up, mysqldump used 168
- database backing up, with backup and migrate module 169
- from calendar 5.x-1.x and date 5.x-1.x to calendar 5.x-2.x and date 5.x-2.x 168
- recipe notes 174

date API module 364

date badge

- about 298-301
- events listing view, creating 299
- recipe notes 301, 302

date browser

- browse_workshops view, editing 151
- recipe notes 153
- view, saving 151, 152

date formats

- about 135-139
- recipe notes 140

date module

- about 364
- URL 140

debugging statement

- Devel module based 277, 278
- for core Drupal 276, 277
- recipe notes 278

default views

- about 42
- code, reviewing 86-89
- comments_recent 43-45
- creating 90
- frontpage view 49, 50
- popular_alltime 57, 58
- popular_recent 63
- taxonomy_term 72-74
- taxonomy view 67-69
- tracker 79- 81

default views, creating

- .info file 90
- about 90
- function in .inc file 90-92
- including, in .module file 92

default views, in Drupal 5 modules

- Acidfree Albums 335
- Activity Stream 335
- Audio 335
- Backlinks 335
- Bibliography Module 335
- Bio 335
- BitTorrent 336
- Buddylist 336
- Buddylist2 336
- Calendar 336
- Case Tracker Work 336
- Competition 336
- Democracy forum 336
- Devel 336
- EasyLink 336
- erp 336, 337
- Event Views 337
- Event Webform 337
- FeedAPI 337
- Flag 337
- FooAggregator RSS aggregator 338
- Forward 338
- Gallerix 338
- Housing Board 338
- Job Search 338
- Kaltura 338
- location (API, module) 338
- Media Portfolio 338
- MLS Module 338
- NGP Links 339

- Nodeforum 339
- Node gift 339
- Node Hierarchy 339
- Nodequeue 339
- NodeReview 339
- Node time tracker 339
- OG Galleries 339
- Organic groups 339
- Pro and Con arguments 339
- Send 339
- Similar Nodes 339
- SimpleFeed 340
- Smartqueue per User 340
- Smartqueues for Organic Groups 340
- Station 340
- Timeline 340
- Ubercart 340
- Update status aggregator 340
- Usernode 340
- userreview 340
- Versus 340
- Video 340
- Views 340
- Views Bonus Pack 341
- Views Bulk Operations 341
- Views Fast Search 341
- Views Mail 341
- Views Multiblock 341
- Webcomic 341
- Workflow Owner 341
- Yet Another Gallery Module 341
- Democracy forum, default views 336**
- dev/null redirect, URL 201**
- Devel, default views 336**
- Devel module**
 - about 201-203, 279, 364
 - recipe notes 203, 280
- Dmitri Gaskin, URL 326**
- Drupal.org 317**
- drupal.org user search, URL 332**
- Drupal 5 7**
- DrupalCon presentation (video), URL 195**
- Drupal Contributions API 5 link 328**
- Drupal Dojo 314**
- Drupal for Facebook, style plugins 356**
- Drupal For Firebug**
 - about 369
 - URL 203
- Drupal groups, URL 332**
- Drupal issue queue, URL 333**
- Drupal modules, URL 332**
- Drupal patch 323, 324**
- Drupal Troubleshooting FAQ, URL 321**
- Druplicon 331**
- dynamic rendering 369**
- dynamic views, style plugins 356**
- E**
 - e-commerce, style plugins 356**
 - EasyLink, default views 336**
 - editablefields, formatters 347**
 - editablefields, style plugins 356**
 - editablefields module 364**
 - about 183
 - alternative, Editeview module 184
 - recipe modules 184
 - URL 183
 - Editeview module**
 - downloading, URL 184
 - Editview, style plugins 356**
 - Education Field, formatters 347**
 - Email Field, formatters 347**
 - Embedded Media Field - Audio, formatters 347**
 - Embedded Media Field - Image, formatters 347**
 - Embedded Media Field - Video, formatters 347**
 - Employment Field, formatters 347**
 - Empty Text, default views**
 - about 74, 75
 - regional categories, list generating 76
 - erp, default views 336, 337**
 - Event Views, default views 337**
 - Event Webform, default views 337**
 - Evoca Media Services - EMS, formatters 347**
 - excerpt 369**
 - Explainfield CCK widget, formatters 347**
 - explore module**
 - default views 258

- Phoneblogz module 258
- views checkboxes module 257
- export, patch name 374**
- exposed filters fieldset**
 - about 144
 - editing 144
 - opening 144
 - recipe notes 145
 - settings 146
- exposed filters fieldset, settings**
 - filter settings default, checked 148
 - filter settings default, option 146
 - filter settings default, unchecked 148
 - Force Single, checked 148
 - Force Single, Unchecked 148
 - lock Operator, checked 148
 - lock operator, option 146
 - lock Operator, unchecked 148
 - recipe notes 148
 - vocabulary, creating for workshop topics 147
- Eye-drop Editor (Deprecated), formatters 348**

F

- fading slideshow 369**
- favicon generator 369**
- featured content Slider 369**
- FeedAPI, default views 337**
- FeedBurner.com 370**
- feed modules, URL 272**
- feed selector argument, frontpage view**
 - default 51
 - option 52
 - settings 51
 - title 52
 - wildcard and wildcard substitution 52
- field copy module**
 - about 364
 - URL 95
- fields, patch name 374**
- FileField, formatters 348**
- filters, patch name 375**
- Firebug**
 - installing 32-34
 - URL 164, 370

- Firefox**
 - about 370
 - installing 32
- Fivestar, formatters 348**
- Flag, default views 337**
- flag module 364**
 - default bookmarks view 249, 250
 - URL 249
- flag module, using for use cases**
 - Fact-Checkers, view creating for 253, 254
 - inappropriate comments, list viewing 253
 - recipe notes 256
 - RSVPs, view 254, 256
- Flickr, formatters 348**
- FLV media player, style plugins 356**
- folding views panes 268**
- FooAggregator RSS aggregator, default views 338**
- FooAggregator RSS aggregator, style plugins 356**
- format_date() function 141**
- formatters**
 - about 100
 - Address field for CCK 344
 - ASIN Field 343
 - Asset 343
 - Asset API 343
 - Assetfield 343
 - Autonode CCK widget 343
 - Body Builder 343
 - Brazilian IDs 343
 - cck_map 345
 - CCK Button 344
 - CCK comment reference 346
 - CCK Currency 346
 - CCK Download Dropdown 344
 - CCK Extras 344
 - CCK Flashcard 345
 - CCK Formatters - Flash 345
 - CCK Formatters - KML 345
 - CCK GMap Address 345
 - CCK IP Address 345
 - CCK Link to Map 344
 - CCK Multimage 345
 - CCK Node Menu 345
 - CCK Node Reference 344
 - CCK Number 344

- CCK Redirection 345
- CCK Table Field 346
- CCK Taxonomy Fields 346
- CCK Taxonomy Super Select Ultra 346
- CCK Text 344
- CCK User Reference 344
- CiviNode And Civinode CCK 346
- Colorpicker 346
- Comic View 346
- Computed Field 346
- Content Construction Kit (CCK) 343
- Content Taxonomy 346
- Craqbox 346
- Date 346
- editablefields 347
- Education Field 347
- Email Field 347
- Embedded Media Field - Audio 347
- Embedded Media Field - Image 347
- Embedded Media Field - Video 347
- Employment Field 347
- Evoca Media Services - EMS 347
- Explainfield CCK widget 347
- Eye-drop Editor (Deprecated) 348
- Field Reference 348
- FileField 348
- Fivestar 348
- Flickr 348
- Fullname field for CCK 345
- Geonames CCK 348
- Giant Scalable Image Viewer 348
- GMap Addons 348
- Guitar 348
- ImageCache 348
- ImageField 349
- Inventory field 349
- jQuery Gallery View for CCK multiple imagefield 349
- jQuery Lightbox 349
- JQZoom 349
- JSNodeLoad 349
- Lightbox2 349
- Link 350
- Location (API, module) 350
- Matrix Field 350
- Media Field - Audiofield 350
- Media Field - Display 350

- Media Field - Videofield 350
- Money CCK field 351
- Multireference CCK field 351
- Namefield 351
- Office Hours 351
- OpenPackage Video 351
- Phone (CCK) 351
- Pollfield 351
- Postal 351
- Preset Widget Options 352
- Private Number 352
- Role Reference 352
- See Map 352
- Shadowbox 352
- SMS Framework 352
- State Reference 352
- Text Field Tags 352
- Thickbox 352
- Timer 353
- Trackfield 353
- Trackfield - Graph 353
- Trackfield - Map 353
- Trackfield - Stats 353
- Ubercart Add to Cart CCK Field 353
- VCalfield 353
- Video Upload 353
- Viewfield 353
- zipcode (CCK) 353
- Zoomify 353
- Forward, default views 338**
- frontpage slideshow, style plugins 356**
- frontpage view, default views**
 - about 49
 - arguments 49
 - basic information fieldset 51
 - duplicate search engine results, removing 55
 - enabling, to show on front page 54, 55
 - features 49
 - feed selector argument 51, 52
 - fields fieldset 51
 - filters 49
 - filters fieldset 52, 53
 - overriding 50
 - page fieldset 49, 51
 - Really Simple Syndication (RSS) feed 49
 - sort 50

- sort criteria fieldset 53, 54
- views RSS module, enabling 50
- Fullname field for CCK, formatters 345**
- FusionCharts, style plugins 356**

G

- Gallerix, default views 338**
- gallery recipe**
 - jQuery update, installing 223
 - Lightbox2, installing 223
 - recipe notes 224
- GCal Events module, URL 143**
- Geonames CCK, formatters 348**
- getID3() module 364**
- getID3 library, URL 242**
- Giant Scalable Image Viewer, formatters 348**
- GMap Addons, formatters 348**
- Gmap module, style plugins 356**
- Google Analytics 370**
- Google Maps**
 - about 225
 - content types, creating 226
 - content types, populating 228
 - Google Map Search 232
 - recipe notes 230, 231
 - tools, setting up 225, 226
 - view, setting up 228
- Google Maps tools 364**
- Google Maps tools, style plugins 356**
- gProximity, style plugins 356**
- Guitar, formatters 348**
- guitar chords, displaying**
 - about 108, 109, 364
 - recipe notes 109

H

- header**
 - adding, to view 23
- header fieldset 24**
- hook_field_formatter() (From the CCK module) 362**
- hook_field_formatter_info() function 103**
- hook_link_alter**
 - output, tweaking 273, 274
 - recipe notes 275

- hook_views_arguments() 361**
- hook_views_arguments_alter() 361**
- hook_views_default_views() 90, 361**
- hook_views_feed_argument() 361**
- hook_views_fusion() (From the Views Fusion module) 362**
- hook_views_post_view() 361**
- hook_views_pre_query() 362**
- hook_views_pre_view() 362**
- hook_views_query_substitution() 362**
- hook_views_style_plugins 361**
- hook_views_tables() 361**
- hook_views_tables_alter() 362**
- hook_views_tabs() 362**
- hook_views_theme_wizard_types() 362**
- Housing Board, default views 338**

I

- iCal feeds**
 - about 165
 - calendar items, enabling 165
 - calendar items, exporting 166, 167
 - recipe notes 167
- iCal module 363**
- image, style plugins 356**
- ImageAPI GD2 364**
- ImageAPI module, URL 218**
- ImageCache module**
 - about 364
 - URL 218
- ImageCache, formatters 348**
- Imagecache module, URL 218**
- ImageField 365**
- ImageField, formatters 349**
- Imageflow, style plugins 356**
- input format filters 24**
- installing, views module 7**
- Instant SQL Formatter**
 - about 370
 - URL 201
- Inventory field, formatters 349**
- IRC**
 - about 330
 - recipe notes 331
- IRC Drupal support**
 - participating, steps 331

issue

- existing issue, responding to 322, 323
- new issue, posting 321, 322
- recipe notes 323

issue page bookmark

- creating 317
- recipe notes 318, 319

J

JavaScript tools 365

jCarousel 365

jCarousel module

- about 306, 307
- URL 306

Job Search, default views 338

jQuery Gallery View for CCK multiple

- imagefield, formatters 349

jQuery Lightbox, formatters 349

jQuery slideshow 370

jQuery Update module

- about 223, 365
- URL 223

JQZoom, formatters 349

JSNodeLoad, formatters 349

JSTools tab module 311

K

Kaltura 370

Kaltura, default views 338

KML module, style plugins 357

L

Lightbox2 365

Lightbox2, formatters 349

Lightbox2 module

- about 223
- recipe notes 224
- URL 223

link 365

Link, formatters 350

location (API, module), default views 338

Location (API, module), formatters 350

M

mageAPI 364

Masquerade 365

Masquerade module

- about 179
- configuring 179, 180
- press release 180, 181
- switch back link 180
- URL 178

Matrix Field, formatters 350

Media Field - Audiofield, formatters 350

Media Field - Display, formatters 350

Media Field - Videofield, formatters 350

media hit content type, creating 115

Media Portfolio, default views 338

menu

- view, adding 25, 26

menu fieldset 25

menu local tasks 59

mini-pie charts, displaying

- pie chart, downloading 131
- pie chart, selecting 132
- pie chart content type, setting up 127, 129
- recipe notes 132
- sIFR, downloading 130
- sIFR, enabling 130
- sIFR, installing 130
- view, setting up 129, 130

MLS Module, default views 338

ModuleInfo module

- about 177, 365
- recipe notes 178

Money CCK field, formatters 351

Multireference CCK field, formatters 351

N

Namefield, formatters 351

new content type, creating 14-16

new issue, link 321

NGP Links, default views 339

Nick Lewis Drupal Hackers Cookbook 314

node cloud, style plugins 357

node comments, style plugins 357

node feed selector argument 50

- Nodeforum, default views** 339
- Node gift, default views** 339
- Node Go To module**
 - about 140, 365
 - URL 140
- Node Hierarchy, default views** 339
- Nodequeue** 370
- Nodequeue, default views** 339
- Node Reference module** 364
 - about 115
 - dropdown press release view, creating 117
 - media hit content type, creating 115
 - media hit content type, modifying 117
 - media hit content type, populating 118
 - modules, enabling 115
 - patching 206
 - press release, adding 116
 - recipe notes 118
- Node reference Patch, URL** 205
- node reference views select, style plugins** 357
- NodeReview, default views** 339
- Node time tracker, default views** 339

O

- Office Hours, formatters** 351
- OG Galleries, default views** 339
- online date converter, URL** 140
- online resources**
 - about 313
 - Drupal Dojo 314
 - Nick Lewis Drupal Hackers Cookbook 314
 - theme developers cheatsheet 313
 - theme development support forum 314
 - theme field 314
 - theme snippets 314
 - theming Drupal 314
 - theming handbook 313
 - theming views 1 314
 - validate CSS 314
- OpenPackage video, formatters** 351
- OpenPackage video, style plugins** 357
- OpenSearch plugins** 330
- organic group galleries, style plugins** 357
- organic groups** 370

- organic groups, default views** 339
- other views patches, patch name** 377
- oyoaha liquid layout, style plugins** 357

P

- pager**
 - formatting, CSS only modifications used 280-282
 - recipe notes 282, 283
- pager preference** 370
- panels** 365
- panels module**
 - about 265
 - feed content, creating 266, 267
 - panels setup 268-271
 - placing, on front page 272
 - recipe notes 272
 - SimplePie program, downloading 265
 - specific views for panels, enabling 268
 - URL 265
 - views, creating 267
- panel style, style plugins** 357
- paranoia** 370
- Paranoia module** 276
- partial, style plugins** 357
- patch**
 - applying 323-325, 373
 - creating 326, 327
 - creating, URL 327
 - recipe notes 325-327
 - URL 326
- patch name**
 - arguments 373
 - breadcrumbs 374
 - comments 374
 - export 374
 - fields 374
 - filters 375
 - others 377
 - other views patches 377
 - RSS 375
 - sort 375
 - SQL 376
 - taxonomy 376
 - theme 376

- Pathauto** 365
- Pathauto, URL** 242
- Phone (CCK), formatters** 351
- PhoneBlogz module**
 - about 258, 365
 - URL 258
- photo gallery**
 - Imagecache, configuring 219, 220
 - photo content type, setting up 218, 219
 - photos, adding 221
 - recipe notes 222
 - view, setting up 221, 222
- PHP tags, guidelines**
 - .info files 275
 - .module files 275
 - .tpl.php files 275
 - block 275
 - content body 275
 - panels custom body content 275
 - template.php 275
 - views custom field value 275
- Pollfield, formatters** 351
- Poormanscron** 366
- Poormanscron module**
 - about 200
 - recipe notes 200
- popular_alltime, default views**
 - default views, overriding 58
 - hits, removing 64
 - menu options 59-62
 - statistics, enabling 58
- popular_recent, default views**
 - about 63
 - blocks, adding to region 63, 64
 - user-submitted snippets 66
- Porter-Stemmer** 370
- Postal, formatters** 351
- posters view , URL** 208
- Preset Widget Options, formatters** 352
- press release**
 - adding 116
- Private Number, formatters** 352
- Pro and Con arguments, default views** 339
- project issue tracking, style plugins** 357
- project usage overview, URL** 332
- proximity search** 232

- PSPad** 370
- PSPad , URL** 203

Q

- quick tabs module**
 - about 311, 366
 - URL 311

R

- Read More Link module** 57, 366
- Read More Tweak module** 56, 57
- Ready to Be Committed.** *See* RTBC
- regenerate module**
 - about 366
 - URL 302
- resources**
 - CVS messages 332
 - drupal.org users, search by name 332
 - Drupal groups 332
 - Drupal modules 332
 - project usage overview 332
 - views 1.7 issue 332
 - views 1.x module snippets 332
 - views documentation 332
 - views t-shirt 332
- reverse node reference**
 - content type, relating with 119
 - recipe notes 120
 - roles 30
- Role Reference, formatters** 352
- RSS, patch name** 375
- RTBC** 327

S

- Scalable Inman Flash Replacement.** *See* sIFR
- search plugin**
 - installing, for contributions API 328, 329
 - recipe notes 329, 330
- See Map, formatters** 352
- Send, default views** 339
- Shadowbox, formatters** 352
- sIFR** 127, 366
- Similar Nodes, default views** 339

- SIMILE timeline** 370
- SIMILE timelines, URL** 156
- SimpleFeed, default views** 340
- SimpleFeed module**
 - about 366
 - URL 265
- SimplePie module**
 - URL 265
- site notes** 370
- Slideshow** 371
- SlideShowPro** 371
- SlideShowPro, style plugins** 357
- slow queries log, URL** 203
- Smartqueue per User, default views** 340
- Smartqueues for Organic Groups, default views** 340
- SMS Framework, formatters** 352
- sort, patch name** 375
- SQL, patch name** 376
- SQL Format extension , URL** 203
- State Reference, formatters** 352
- Station, default views** 340
- Statistics** 366
- style plugins**
 - action views 355
 - active template 355
 - activity stream 355
 - ad views 355
 - AJAX views 355
 - asset API 355
 - bibliography module 355
 - calendar 355
 - carousel 355
 - conference organizing 355
 - date 356
 - Drupal for Facebook 356
 - dynamic views 356
 - e-commerce 356
 - editablefields 356
 - Editview 356
 - FLV media player 356
 - FooAggregator RSS aggregator 356
 - frontpage slideshow 356
 - FusionCharts 356
 - Gmap module 356
 - Google Maps tools 356
 - gProximity 356
 - image 356
 - Imageflow 356
 - KML module 357
 - node cloud 357
 - node comments 357
 - node reference views select 357
 - OpenPackage video 357
 - organic group galleries 357
 - oyoaha liquid layout 357
 - panel style, style plugins 357
 - partial 357
 - project issue tracking 357
 - SlideShowPro 357
 - timeline 357
 - view of views 359
 - views 357
 - views_multiblock 359
 - views_slideshow 359
 - views bonus pack 357
 - Views Bulk Operations (VBO) 358
 - views calc 358
 - views carousel 358
 - views Datasource 358
 - views fast search 358
 - views Group-By pack 358
 - views JSON 359
 - views mail 359
 - views many node export 357
 - views node feed 359
 - views podcast 359
 - views popup 359
 - views rotator 359
 - views style plugin options 359
 - views tabs 359
 - views tagadelic 359
 - views ticker 360
 - XML sitemap 360
 - XSPF playlist 360
 - Yet Another Gallery module 360
- Submit More module, URL** 143
- summary views**
 - argument, deleting 150
 - datestamp workshop date argument, adding 149
 - editing 149

- recipe notes 151
- view, saving 150
- wildcard 149
- swim groups, block creating** 27-29
- swim groups page view**
 - filters, selecting 19-21
 - new content type (swim group),
 - creating 14-16
 - view, creating 17, 18

T

- tabs module**
 - about 309, 310
 - recipe notes 311
 - URL 309
- tandem views panes** 268
- taxonomy 371**
- taxonomy, patch name** 376
- taxonomy_term, default views**
 - about 72-74
 - Empty Text option 74, 75
 - regional categories, list generating 76, 77
- taxonomy access control lite** 371
- taxonomy hide** 371
- taxonomy manager** 366, 371
- taxonomy redirect** 371
- taxonomy view**
 - about 67-69
 - overriding 69
- teaser list** 50
- template.php** 139
- Text Field Tags, formatters** 352
- theme, patch name** 376
- theme_blocks** 287
- theme_comment** 287
- theme_feed_icon** 287
- theme_function_name** 285
- theme_node** 287
- theme_node_list** 287
- theme_page** 287
- theme_pager** 288
- theme_pager function**
 - calling 285, 286
 - overriding 283, 284
 - recipe notes 284-286

- theme_table_sort_image** 288
- theme_user_picture** 288
- theme_views_bonus_export_doc** 288
- theme_views_bonus_export_text** 288
- theme_views_edit_view** 288
- theme_views_filters** 288
- theme_views_summary** 288
- theme_views_view_list** 288
- theme_views_view_table** 288
- theme_views_view function**
 - overriding 288, 290
 - recipe notes 290
- theme_xml_icon** 288
- themeable function names**
 - displaying, in source code 263
 - recipe notes 264
- themeable functions**
 - about 262
 - recipe notes 263
- theme developers cheatsheet** 313
- theme field** 314
- themes_views_view** 288
- theme snippets** 314
- theming Drupal** 314
- theming handbook** 313
- theming views 1** 314
- Thickbox, formatters** 352
- This theme function** 287, 288
- timeline, default views** 340
- timeline, style plugins** 357
- timeline module** 366
 - about 153, 154
 - controls, enabling 156
 - recipe notes 156
 - timeline view, editing 154
 - URL 153
 - view, saving 155
- timelines, URL** 156
- timeline widget** 371
 - URL 153
- Timer, formatters** 353
- token** 366
- token, URL** 242
- total item count**
 - about 290, 291
 - recipe notes 291

- tracker** 371
- tracker, default views**
 - about 79-81
 - making sortable 82
 - view, integrating with tracker
 - module 84, 85
- Trackfield, formatters** 353
- Trackfield - Graph, formatters** 353
- Trackfield - Map, formatters** 353
- Trackfield - Stats, formatters** 353
- U**
- Ubercart** 366
- Ubercart, default views** 340
- Ubercart Add to Cart CCK Field, formatters** 353
- UI** 11
- unified context diff** 327
- Update status aggregator, default views** 340
- Update Status module, URL** 174
- User Interface.** *See* UI
- Usernode, default views** 340
- userreview, default views** 340

V

- validate CSS** 314
- VBO module**
 - about 185
 - default views 185
 - filters field set 186
 - installing 185
 - recipe notes 188, 189
 - view, saving 186
- VCalfield, formatters** 353
- Versus, default views** 340
- Video, default views** 340
- Video Upload, formatters** 353
- view**
 - adding, to menu 25, 26
 - blocks 27, 29
 - content, creating 14-16
 - embedding, in theme template 121-125
 - jQuery command, running 37
 - recipe notes 126

- view alias** 371
- Viewfield** 366
- Viewfield, formatters** 353
- Viewfield module**
 - content setup 110, 111
 - modules, installing 110
 - recipe notes 113-115
 - taxonomy 110
 - view, associating with content
 - item 112, 113
 - views 110
- view of views, style plugins** 359
- views, default views** 340
- Views: Many Node Export** 370, 371
- views, style plugins** 357
- views-related URLs, on YOURSITE**
 - administer views 13
 - view, adding 13
 - view, importing 13
 - views help 13
 - views modules, enabling 13
 - views permissions 13
 - views tools 13
 - views UI help 13
- views_multiblock, style plugins** 359
- views_slideshow, style plugins** 359
- views_style_plugins** 329
- views 1.7 issue, URL** 332
- views 1.x module snippets, URL** 332
- views 1 patches list, URL** 373
- views body field issue.** *See* body field issue
 - body field, copying 98
 - body field issue, demonstrating 96-98
 - recipe notes 99
 - view, modifying 98
- Views Bonus Export** 367
- Views Bonus Pack, default views** 341
- views bonus pack, style plugins** 357
- views bonus pack module**
 - about 216, 367
 - recipe notes 217
 - URL 215, 218
 - view, saving 216, 217
- Views Bulk Operations (VBO)** 367
- Views Bulk Operations (VBO), style plugins** 358

- Views Bulk Operations, default views** 341
- views calc** 367
- views calc, style plugins** 358
- views calculations**
 - about 240, 241
- views carousel, style plugins** 358
- views carousel module**
 - about 306, 307
 - URL 308, 309
- views checkboxes module**
 - about 257, 367
 - URL 257
- views custom field module**
 - about 232-235, 367
 - recipe notes 196, 238, 239
 - swim_groups view, cloning 195
 - URL 195, 232
 - view, adding 236
 - view, saving 236
- views dashboard, code approach**
 - about 309, 310
 - recipes notes 311
- views dashboard, module approach**
 - about 311, 312
 - recipe notes 313
- views Datasource, style plugins** 358
- views date range filter module** 367
 - editing 145
 - recipe notes 146
 - URL 145
- views documentation, URL** 332
- views edit screen, fieldsets** 36
- Views Fast Search, default views** 341
- views fast search, style plugins** 358
- views fast search module**
 - alternative imported view, using 210, 211, 367
 - default views, using 209
 - recipe notes 212
 - search block, replacing 211
 - setting up 209
 - URL 209, 212
- views fusion** 205, 367
 - configuring 207, 208
 - documentation 209
 - installing 207
 - Nodereference module, patching 206
 - views setup 206, 207
- views Group-By pack, style plugins** 358
- views hooks**
 - frequently used 361
 - from other modules 362
 - not used frequently 361, 362
- views issue queue**
 - recipe notes 320, 321
 - searching 319
 - searching, advanced search 319
 - searching, basic search 319
 - searching, options 320
- views JSON, style plugins** 359
- Views Mail, default views** 341
- views mail, style plugins** 359
- views many node export, style plugins** 357
- views module** 367
 - <DRUPALHOME>/sites/all/modules
 - directory, creating 8
 - <DRUPALHOME>/sites/all/modules
 - directory, finding 8
 - about 7
 - downloading 9
 - enabling 10
 - uncompressing 10
- Views Multiblock, default views** 341
- views node feed, style plugins** 359
- view source chart** 371
- view source chart add-on, URL** 264
- views podcast, style plugins** 359
- views popup, style plugins** 359
- views popup module**
 - about 157, 367
 - file 158, 159
 - opening 157
 - recipe notes 159
 - URL 157
 - view, saving 158
- views queries**
 - formatting 201, 202, 203
- views rotator** 371
- views rotator, style plugins** 359
- views RSS** 368
- Views slideshow** 371

- views style plugin options,**
 - style plugins 359**
- views t-shirt, URL 332**
- views tab 311, 371**
- views tabs, style plugins 359**
- views tagadelic, style plugins 359**
- views taxargs 371**
- views theme wizard module**
 - about 292, 367**
 - FAQs 293, 294, 295**
 - recipe notes 296, 297**
- views ticker, style plugins 360**
- views UI module 11**
 - recipe notes 181**
 - URL 178**
- Views UI permissions 178, 368**
- Views UI sort module**
 - about 368**
 - recipe notes 183**
- views worksheets**
 - preparing 38**
- view title count 371**

W

- Webcomic, default views 341**
- Web Developer Toolbar, URL 264**

- website snapshots**
 - content type, creating 100**
 - link field, adding 101**
 - modules, installing 100**
 - recipe notes 103-105**
 - view, creating 101, 102**
- Websnapr, URL 103**
- Workflow Owner, default views 341**

X

- XML sitemap, style plugins 360**
- XSPF playlist, style plugins 360**

Y

- Yet Another Gallery Module,**
 - default views 341**
- Yet Another Gallery module,**
 - style plugins 360**
- YouTube video bar**
 - Google Video Bar Wizard , URL 213, 214**
 - recipe notes 214, 215**

Z

- zipcode (CCK), formatters 353**
- Zoomify, formatters 353**



**Thank you for buying
Drupal 5 Views Recipes**

Packt Open Source Project Royalties

When we sell a book written on an Open Source project, we pay a royalty directly to that project. Therefore by purchasing *Drupal 5 Views Recipes*, Packt will have given some of the money received to the Drupal project.

In the long term, we see ourselves and you – customers and readers of our books – as part of the Open Source ecosystem, providing sustainable revenue for the projects we publish on. Our aim at Packt is to establish publishing royalties as an essential part of the service and support a business model that sustains Open Source.

If you're working with an Open Source project that you would like us to publish on, and subsequently pay royalties to, please get in touch with us.

Writing for Packt

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

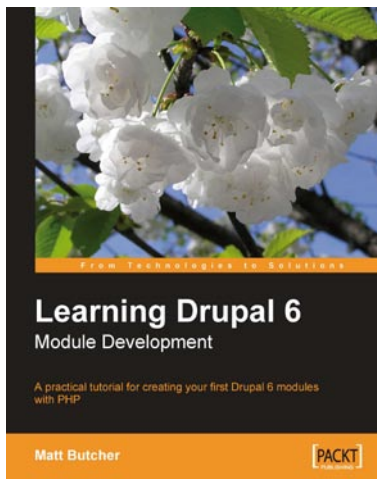
We're not just looking for published authors; if you have strong technical skills but no writing experience, our experienced editors can help you develop a writing career, or simply get some additional reward for your expertise.

About Packt Publishing

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

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

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



Learning Drupal 6 Module Development

ISBN: 978-1-847194-44-2

Paperback: 310 pages

A practical tutorial for creating your first Drupal 6 modules with PHP

1. Specifically written for Drupal 6 development
2. Program your own Drupal modules
3. No experience of Drupal development required
4. Know Drupal 5? Learn what's new in Drupal 6
5. Integrate AJAX functionality with the jQuery library
6. Packt donates a percentage of every book sold to the Drupal foundation



Drupal 6 Themes

ISBN: 978-1-847195-66-1

Paperback: 291 pages

Create new themes for your Drupal 6 site with clean layout and powerful CSS styling

1. Learn to create new Drupal 6 themes
2. No experience of Drupal theming required
3. Techniques and tools for creating and modifying themes
4. A complete guide to the system's themeable elements

Please check www.PacktPub.com for information on our titles