



Joomla!

(templates)

Angie Radtke



Joomla!® Templates

This page intentionally left blank

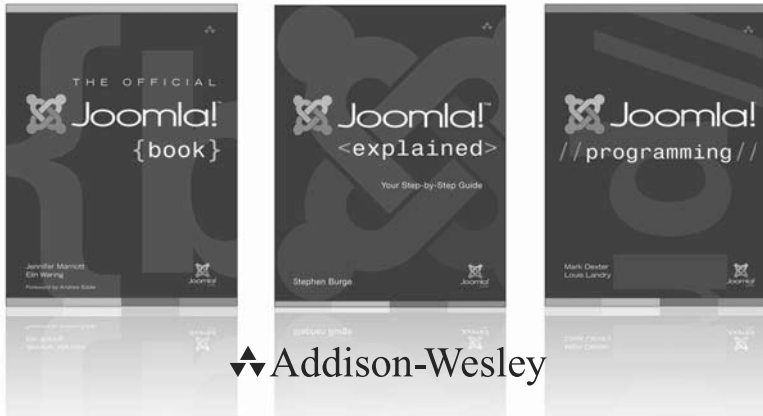
Joomla!® Templates

Angie Radtke

◆Addison-Wesley

Upper Saddle River, NJ • Boston • Indianapolis • San Francisco
New York • Toronto • Montreal • London • Munich • Paris • Madrid
Capetown • Sydney • Tokyo • Singapore • Mexico City

Joomla! Press



Addison-Wesley

Visit informit.com/joomlapress for a complete list of available publications.



The mission of Joomla! Press is to enhance the Joomla! experience by providing useful, well-written, and engaging publications for all segments of the Joomla! Community from beginning users to platform developers. Titles in Joomla! Press are authored by the leading experts and contributors in the community.



Make sure to connect with us!
informit.com/socialconnect



informIT.com
THE TRUSTED TECHNOLOGY LEARNING SOURCE

Safari
Books Online

A Message from Open Source Matters

Since Joomla! launched in September 2005, it has grown to become one of the most popular content management systems in the world. As this book goes to press in July 2012, Joomla! has been downloaded over 32,000,000 times and provides support for 64 different languages. Joomla! has received multiple awards, and estimates indicate that approximately 2.8% of all Internet Web sites are using Joomla!.

The key to Joomla!'s success has always been the help and contributions freely given by a large and diverse group of volunteers from all over the world. The Joomla! project isn't backed by venture capital firms, and it isn't led by a single individual or corporation. It is volunteers who write the code and then test it, translate it, document it, support it, extend it, promote it, and share it.

Volunteers are also continually planning and organizing events all over the world where people come together to learn, connect, and share about Joomla!. These events include hundreds of local user groups, as well as national and international conferences. The first Joomla! World Conference will take place in November 2012 in San Jose, California (go to <http://conference.joomla.org> for more information).

Work is underway on many improvements and new ideas aimed at keeping Joomla! on a path of continued growth and innovation. Our community is open to all. If the idea of working alongside a diverse group of bright and passionate volunteers from all over the world who are helping to make Joomla! better sounds fun and rewarding to you, then I invite you to join us. To learn more, please go to <http://www.joomla.org>.

Best regards,

Paul Orwig
President, Open Source Matters

Open Source Matters is the nonprofit organization that provides legal, financial, and organizational support for the Joomla! project.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The author and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

The publisher offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact:

U.S. Corporate and Government Sales
(800) 382-3419
corpsales@pearsontechgroup.com

For sales outside the United States, please contact:

International Sales
international@pearson.com

Visit us on the Web: informit.com/aw

Library of Congress Cataloging-in-Publication Data

Radtke, Angie.

Joomla! templates / Angie Radtke.
p. cm.

Includes bibliographical references and index.

ISBN 978-0-321-82731-9 (pbk. : alk. paper)

1. Joomla! (Computer file) 2. Web sites—Authoring programs. 3. Web site development. I. Title.

TK5105.8883.R32 2013

006.7'8—dc23

2012017878

Copyright © 2013 Pearson Education, Inc.

All rights reserved. Printed in the United States of America. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. To obtain permission to use material from this work, please submit a written request to Pearson Education, Inc., Permissions Department, One Lake Street, Upper Saddle River, New Jersey 07458, or you may fax your request to (201) 236-3290.

ISBN-13: 978-0-321-82731-9

ISBN-10: 0-321-82731-7

Text printed in the United States on recycled paper at RR Donnelley in Crawfordsville, Indiana.

First printing, July 2012

Editor-in-Chief

Mark L. Taub

Executive Editor

Debra Williams Cauley

Development Editor

Songlin Qiu

Managing Editor

John Fuller

Project Editor

Elizabeth Ryan

Packager

Kim Arney

Copy Editor

Carol Lallier

Indexer

Richard Evans

Proofreader

Diane Freed

Technical Reviewer

Andrea Tarr

Editorial Assistant

Kim Boedigheimer

Compositor

Kim Arney

Contents

Introduction xvii

Acknowledgments xxi

About the Author xxiii

1 The Basis: Designing the Content and Visual Concept 1

It All Starts with the Structure 1

Recognizing User Expectations 2

Page Layout—Visually Structuring Content 3

Designing with Grids 3

Implementation 5

Push to Front Principle 7

The Graphical Layout—Visual Appearance Matters 7

Colors—A Central Element 8

Designing the Navigation—The Core of the Design 12

Content Design—To Make It Fun to Read 12

Font Design—We Do Not Have Many Options 13

Fixed and Fluid Layouts 16

2 Accessibility—What Is It? 19

The Legal Basis 20

Visual Impairment 21

Initial Situation and Findings 21

Technical Aids 22

What Can We Do? 28

Motor Disabilities 29

Initial Situation and Findings 29

Technical Aids 30

What Can We Do? 30

Deafness 31

Initial Situation and Findings 31

Technical Assistance 31

What Can We Do? 31

Learning Disabilities 31

Initial Situation and Findings 32

What Can We Do? 32

Seniors	33
Initial Situation and Findings	33
What Can We Do?	34
3 CSS and HTML—Getting the Basic Structure into Shape	35
A Few Words about the History	35
Which Version of HTML Should I Use?	36
HTML 4.01 and XHTML 1.0	36
HTML5	37
The Basic HTML Structure	38
A Brief Introduction to CSS	38
Adding CSS Statements	38
CSS Selectors	40
Inheritance	44
Using Multiple Classes Together	44
Positioning and Box Model	47
CSS Hacks and Browser Problems	52
Conditional Comments	52
The * Hack	52
Internet Explorer Again: hasLayout	53
CSS Tuning	54
CSS3—A Brief Overview	55
Vendor Prefixes	55
Overview of the Three Most Useful CSS Statements	56
border-radius	56
box-shadow	56
linear-gradient	57
4 Responsive Web Design	59
But How Does It Work?	59
CSS3 Media Queries	60
Option 1—Integration into the Main Stylesheet	60
Option 2—Integrating Separate Stylesheets	61
Adapting Graphics and Videos	61
Using HTML5 Apps	62

- 5 PHP and Joomla! 63**
 - Integrating PHP **63**
 - Comments **65**
 - echo **65**
 - Outputting Strings **65**
 - Outputting Variable Values **65**
 - Conditions: if Statements **66**
 - if Statement **66**
 - else Statement **68**
 - For Pros: Accessing Objects and Their Values **68**
 - Parameter Basics **68**
 - Using Parameters **69**

- 6 MooTools 71**
 - Why MooTools? **72**
 - MooTools Quick Start—Dollar Functions and Events **73**
 - The MooTools Core in Action **74**
 - The Class System **76**
 - The MooTools Principle **79**
 - Related Links **81**

- 7 Tools 83**
 - HTML Validator and CSS Validator **83**
 - Web Developer Toolbar **84**
 - Firebug **85**
 - Helpful Tools for Accessibility **86**
 - Colour Contrast Analyser **86**
 - Accessibility Extensions for Internet Explorer and Mozilla Firefox **87**
 - Wave **88**
 - WCAG 2 Checker of the University of Toronto **89**
 - Tilt 3D **89**

- 8 Now for the Details: A First Look at Templates 91**
 - Atomic **91**
 - beez_20 and beez5 **91**
 - The Template Manager: Styles **92**

- The Template Manager: Templates **94**
 - The Template Preview **95**
 - Template Details **97**
 - Installing Templates **99**
- 9** The Underlying Structure **101**
 - The Heart of the Matter, the index.php **102**
 - The css Folder **102**
 - templateDetails.xml **103**
 - The images Folder **103**
 - The html Folder **103**
 - The javascript Folder **103**
 - The language Folder **103**
 - component.php **103**
 - error.php **104**
 - template_thumbnail.png and template_preview.png **104**
 - favicon.ico **104**
 - The fonts Folder **105**
 - The index.html **105**
- 10** The index.php: The Heart of the Matter **107**
 - The Document Head **107**
 - Safety First: Security **107**
 - Which Document Type? **108**
 - HTML Language Indicator **108**
 - jdoc: include type:head **109**
 - Integrating CSS and JavaScript **112**
 - Integrating MooTools **113**
 - Reading Direction from Right to Left **113**
 - And Off We Go: The Body **114**
- 11** The XML File and the Template Parameters **117**
 - templateDetails.xml: General Information **117**
 - Customizing Template Names **119**
 - Integrating Files and Folders **123**
 - Defining Module Positions **123**
 - The Language Files **124**

- Template Parameters: config **124**
 - Adding Your Own Form Fields and Accessing Them **126**
 - Adding Form Elements **129**
- 12 The Language Files 135**
 - How Joomla! Translates Constants to Multiple Languages **135**
 - Adding Your Own Languages **136**
 - Joomla! Conventions for Using Language Strings **137**
 - Language Files in index.php Using the Examples of Skip Links **137**
- 13 Modules—Dynamics within the Presentation 139**
 - jquery:include **139**
 - The name Attribute **140**
 - The style Attribute and the Default Styles **144**
 - Beez Styles **146**
 - Integrating the Module Flexibly into the Layout **149**
 - Adapting ID and CSS **150**
 - The Module Class Suffix **151**
 - The Menu Module **155**
 - Horizontal Navigation with Subnavigation **156**
 - Folded Out Menu **157**
 - Styling Individual Menu Items via Individual Classes **159**
 - Link Image **159**
 - Allocating Individual Link Titles **160**
- 14 Designing Default Output Individually 161**
 - Inspecting the Default Output **161**
 - The Page Class Suffix **162**
 - Template Overrides **165**
 - Model-View-Controller **166**
 - Shifting Output to the Template **168**
 - Adapting Output **169**
 - New—A View with Different Output **170**

- 15** The System Template: Adapting and Modifying Output **173**
 - System Notices **173**
 - Integrating the Messages into the index.php File **176**
 - Adapting the Language **176**
 - Error Messages **176**
 - Replacing System Graphics **179**
 - component.php and How to Do Magic with It **179**
 - Component View with Search Engine–Friendly URLs **181**
 - The component.php File as the Basis for Custom Views **182**
 - offline.php **183**

- 16** Advanced Template Customization Tricks **185**
 - When the Reading Direction Changes: Right-to-Left Languages **185**
 - Integrating RTL CSS **186**
 - Testing RTL Mode **187**
 - PHP Browser Switch **187**
 - PHP Tricks **189**
 - Structuring the Homepage Differently—Access to the Views **190**
 - Outputting the Current Date with PHP **190**

- 17** The Default Templates and Their Features **193**
 - beez_20 and beez5 Templates **193**
 - Accessibility in General **194**
 - beez_20: Selectable Design **196**
 - Position of the Navigation Column **197**
 - JavaScript and WAI-ARIA **199**
 - beez5: Using HTML5 **205**
 - Atomic Template **207**

- 18** Practical Implementation **211**
 - Concept of the Beez Templates **212**

- 19 Step by Step to a New Layout 217**
 - Step 1: Positioning the Navigation **218**
 - Problem **218**
 - Action **218**
 - Step 2: Filling the Center Column with Content **218**
 - Problem **218**
 - Action **218**
 - Optimizing Step 2: More Meaningful Names for Module Positions **219**
 - Step 3: Adjusting the Number of Articles **223**
 - Problem **223**
 - Action **223**
 - Step 4: Visually Designing the Header **223**
 - Problem **223**
 - Actions **223**
 - Result **234**
 - Step 5: Integrating the Module Position for the Header Picture **235**
 - Problem **235**
 - Action **235**
 - Step 6: Adapting the Footer **239**
 - Problem **239**
 - Action **240**
 - Step 7: Adapting the Minimum Height of Content **240**
 - Problem **240**
 - Step 8: The First Tests **241**
 - Font Enlargement **241**
 - Keyboard Operation **242**
 - Browser Check **243**
 - Step 9: Customizing Typography **245**
 - Problem 1 **245**
 - Action 1 **245**
 - Problem 2 **246**
 - Action 2 **246**
 - Step 10: Formatting Module Headings **246**
 - Problem **246**
 - Action **246**

Step 11: Assigning the Background Image to the Homepage Article	249
Step 12: Final Tests	253
Validating CSS	253
Validating HTML	253
Browser Check	255
Accessibility Checks	257
20 Integrating Custom Features	259
The Header Image—A Background Image?	259
Editing Module Content	261
Adapting CSS	262
Background Images in the Module's Own HTML	264
Browser Check	264
Using HTML5 Effectively	265
Adding the HTML5 Overrides	267
Adapting index.php	269
Adding the JavaScript File to Deal with Internet Explorer	271
Adapting CSS	271
21 Final Tasks: Fine-Tuning and Creating an Installable Zip Archive	273
Fine-Tuning	273
Creating a Print Stylesheet	273
Adjusting error.php and offline.php	274
Right-to-Left View	274
Removing Superfluous Files	274
Creating Previews	274
Changing Favicon	275
Optimizing index.php	275
Adapting the XML File	277
Creating a Zip Archive	278
Appendix	279
Useful Links	279
Joomla!	279
Assistive Technologies	279
CSS	279

HTML5	280
Design	280
Typography	280
Colors	281
Icons	281
JavaScript	281
WAI ARIA	281
Checker Tools	281
Helpful Functions	282
CSS Classes Used and Their Elements	283
Templates	283
Components	287
Modules	305
Index	315

This page intentionally left blank

Introduction

Joomla! is one of the best known Open Source content management systems with many hundreds of thousands of applications in the most varied areas of use. It offers the best possible conditions for implementing a comprehensive and accessible Web presence. Thousands of extensions for almost any purpose are freely available. The developer and user community is huge. On the Internet you can find many different platforms for exchanging information with other users and developers. That's an advantage you should not underestimate! But a Web site without individual design is inconceivable. After all, it's not just the content that makes a Web site truly unique; above all, it's the individual design. This design is the job of the Joomla! templates. In addition to the design aspect, they are also responsible for structuring the content. They create the framework and are basically a template for the content. So they control not only what something looks like but also where the content is located within the document. Joomla! template designers are responsible not only for the design but also for the architecture of the information. When designing a Web site, you need to take into account all requirements of the client as well as the expectations of the visitors.

A small, but important part of these requirements is accessibility. With Joomla!, it's really easy to create accessible Web pages.

To develop Joomla! templates, you need some knowledge of different areas of Web technology, much of which has little to do with Joomla! itself. In our time of increasingly manifold technical possibilities, it is difficult to be an expert in all available Web technologies, so we tend to specialize in certain areas. For instance, you have the front-end developer who knows all there is to know about HTML and Cascading Style Sheets (CSS), the designer who can use Photoshop with all its functions, the PHP specialist, and the JavaScript expert. To develop Joomla! templates, you need some of all this specialized knowledge.

Why This Book Is Unique

This book does not replace a specialized reference work on usability, CSS design, information architecture, PHP, JavaScript, accessibility, or HTML5, but it discusses certain aspects of these topics and others. The aim of this book is to give you the required basic knowledge you need to develop Joomla! templates.

I offer you a readily comprehensible guide that makes it easier for Web designers and programmers to develop their own Joomla! template by working through practical examples. All topics mentioned in this book are condensed to their essence, which was

particularly hard to achieve because I could easily have written whole books on each topic. I hope I succeeded and that you find my book helpful.

How This Book Is Organized

My first aim is to show you how Joomla! templates are constructed and how you can create an accessible, standards-compliant template by using the technical possibilities offered by Joomla! in combination with the most modern forms of technology.

In the opening chapters of the book, you will find general basic information on the individual Web technologies, comments on design, and a list of helpful tools. In principle, the things I describe in this part are the basic requirements you need to build a template in the first place. They are meant to help you get started with these topics. If you are a Web designer, you will probably already be familiar with most of the information contained in this part. In that case, you can move straight on to the second part.

The subsequent chapters discuss the technical background of constructing templates. Using concrete examples, I show you the technical options and internal interrelations.

The final chapters are more practical and presented in the form of a workshop. I demonstrate how to turn a template created in Photoshop to a Joomla! template, step by step.

As happens with any vigorous, ongoing project, Joomla! is always evolving. This book contains the most recent information available at the time of publication but see informIT.com/title/0321827317 for bonus chapters on future releases.

What You Need to Know Before Using This Book

This book is not a “click instruction” but aims to explain contexts and encourage working independently. It is not a CSS book either, although CSS is an important component in building your Joomla! template and is discussed frequently. Photoshop, JavaScript, and PHP are also important tools for your Web design. This book doesn’t provide tutorials on these tools, so you may find it helpful to consult textbooks on these topics.

When you start reading this book, keep these hints in mind.

- As an Open Source project, Joomla! is subject to constant changes. In some chapters I refer to code by specific line numbers. It may well be that these lines move about a bit during the development, because code sections are inserted or removed. I added the references anyway to help you get close to the right place. So if you look something up and it’s not on the specified line number, please just look a bit above or below it.

The potential changes that affect the line references usually result from new features being integrated or old ones removed in different Joomla! versions. Most of what I describe here should apply to older versions as well, and major changes are not expected in the newer versions. But please do not be surprised if there are some slight differences.

- To get the most out of the book, you should install Joomla! (with the sample data that comes with it) onto a Web server. You need to have full access to the file system. The best option is to install a local Web server on your computer, such as XAMPP (www.apachefriends.org/en/xampp.html). This is especially important by the time you get to Chapter 8, “Now for the Details: A First Look at Templates.”
- You will also find it very helpful if you can work with Firefox and install the extension Firebug, which will make your work much easier. You can find out what Firebug is and where to get it when you get to Chapter 7, “Tools.”

Joomla! templates is a wide topic. I have tried very hard to cover all the important points in sufficient detail, but I may have missed something. If you do notice anything, I would be grateful if you could get in touch. Just e-mail me at a.radtke@derauftritt.de. I hope you have fun reading and working your way through this book!

This page intentionally left blank

Acknowledgments

In 2010, Joomla! was at the center of my creative activities. I spent much time using—and greatly enjoying—Joomla!. Working with the templates and the default output has helped me both professionally and personally. I have learned so much and am happy with the outcome of my work. This book was created as a result of it.

Those people with whom I spent a large amount of my time chatting on Skype were also involved. We worked out concepts, made plans, and contrived specific solutions. This includes the always prepared Jean-Marie Simonet, whose commitment I can only admire. Then there is Andrea Tarr, who turned out to be a fellow campaigner for accessibility. There is also Elin Waring, who never seems to sleep. My gratitude goes to Mark Dexter, who always remains calm, and Bill Richardson, the good spirit of bug tracking, who sometimes had to test my patches twice. And I should not forget to mention my “rubber ducky,” Sam Moffat, who was able to solve my problems just by listening to me (maybe he has magic powers?). I also owe thanks to Mahmood and Ofer, who took care of the RTL-CSS of the templates, and to Henk van Cann for listening, to Ian MacLennan, Andrew Eddy, Louis Landry, Jennifer Marriott, and many others.

Special thanks to my colleague and friend Michael Charlier, who supported me with many helpful tips and important advice. I would also particularly like to thank my friend Biggi Mestmäcker for having the *patience of a saint, for providing the linguistic fine polish, and for the fact that she still answers the phone when I call*. Also I would like to thank the editor of the German edition, Boris Karnikowski, for his encouraging words and his trust in me.

I am very happy that my book has also been translated into English and would like to thank the U.S. team at Pearson for their wonderful work. Special thanks are due to Almut Dworak, the translator, whose valuable feedback has certainly helped improve this edition significantly.

But my biggest thanks go to my family. I am very grateful to my husband, Markus Kummer, for having strong nerves and quietly suffering my temper. I would like to apologize to my daughters, Malou and Joelle, for not listening to them and sometimes not being quite sure what I just agreed to. And last but not least, I want to thank my parents-in-law, who always made sure that I also got a bit of the Sunday roast.

This page intentionally left blank

About the Author

Angie Radtke, along with her colleagues at her communications agency, Der Auftritt (www.der-auftritt.de), has been conceiving, designing, and implementing targeted communications solutions since 1999, primarily in the areas of Internet and print. She specializes in marketing-oriented, accessible Web presences and tends to use the open source content management system Joomla!, depending on the customer's wishes.

Appealing design, accessibility, and use of a content management system are not mutually exclusive, and therein lies the basis of Radtke's work. She invests a lot of time and energy in further developing Joomla!. Radtke was actively involved in promoting accessibility in the previous version, Mambo, and her dedication continues today. She developed the two default templates, Beez 2.0 and Beez 5, and she sees herself as an interface between Joomla!'s program logic and its actual output of contents.

Radtke is increasingly involved in passing on her knowledge to others—for example, in training sessions, presentations, and workshops on Joomla! and through accessible Web design. In 2006, she and coauthor Michael Charlier published *Barrierefreies Webdesign: Zugängliche Websites attraktiv gestalten* (München: Addison-Wesley), a book on designing attractive, accessible Web sites.

Angie Radtke is married, has two children, and lives in Bonn, Germany.

This page intentionally left blank

Now for the Details: A First Look at Templates

You are using Joomla! now and would like to know more about its template functions. This chapter gives you an overview of the structure of the templates—knowledge that will help you develop your own individualized template based on the default templates.

Templates determine the general structure of a page. Apart from the design, they determine where the content is within the document, where and when certain modules are shown or hidden, whether to use your own error pages, and which HTML version you should use.

The standard version of Joomla! currently contains three templates for the front end and two for the back end. The frontend templates are two versions of Beez and one Atomic template. In the back end are the standard template Bluestork and the accessible template Hathor by Andrea Tarr. We are going to ignore the backend templates for now.

First and foremost, the frontend templates should be captivating. You can adapt templates according to how you want your Web site to look and feel to the outside world. All templates differ not just in their visual design but also in their range of technical functions. Here is a brief overview of the frontend templates.

Atomic

The focus of the Atomic template is on using the Cascading Style Sheet (CSS) framework Blueprint. Prepared CSS classes help you create complex layouts. If you select this template in the back end and look at the page, it initially seems to have no design. It only appears this way, though, because it is not designed to work with the current sample data. Once you adapt it, you will have a very nice design. More on Atomic a bit later.

beez_20 and beez5

A template usually contains more than you can see at first glance. Apart from the design, it distinguishes itself by how it is implemented in technical terms.

You may be familiar with the Beez template version 1.5. When I created it, I wanted to build a standards-conforming, easily accessible and adaptable template. I chose to use the color purple to make it obvious that you were meant to customize the template to fix your style rather than use it as it was. I hoped that many designers would use the code, modify it creatively, and make it available for free use. I was counting on a multitude of new templates. Sadly, this has not happened. Many users did not understand how to modify the template, and many others did use the code but did not publish their templates.

I made another mistake in not communicating clearly what I had in mind. The output was structured in such a way that almost any design could be achieved with it, simply and without complications. Easy modifications could also be made in the CCS code. I have kept to this principle with each new version of Beez, while making some important changes. There is now more accessible JavaScript. `beez5` has a small portion of HTML5, and `beez_20` manages without template overrides because the default output has been adapted to the output of the old Beez templates, so overrides are not required.

In `beez5` you will find HTML5 code in the overrides.

The Template Manager: Styles

The Template Manager in the back end has the task of managing existing templates in an organized way. It shows you the installed and available frontend and backend templates in a clear list. You can find the template manager in the back end at `Extensions → Template Manager`.

If you open the Template Manager, you will first notice the selected (in-use) styles of the installed templates, as shown in Figure 8.1. Styles are variations of the same template.

The screenshot shows the Joomla! Administration interface for the Template Manager: Styles. The page has a navigation bar with 'Administration' and 'Joomla!' logos. Below the navigation bar, there are tabs for 'Styles' and 'Templates'. The main content area contains a table of installed styles. The table has columns for 'Style', 'Location', 'Template', 'Default', 'Assigned', and 'ID'. The styles listed are:

Style	Location	Template	Default	Assigned	ID
<input type="checkbox"/> Atomic - Default	Site	atomic	☆	☑	3
<input type="checkbox"/> Beez5 - Default-Fruit Shop	Site	beez5	☆	☑	6
<input type="checkbox"/> Beez2 - Parks Site	Site	beez_20	☆	☑	114
<input type="checkbox"/> Beez2 - Default	Site	beez_20	★	☑	4
<input type="checkbox"/> Bluestork - Default	Administrator	bluestork	★		2
<input type="checkbox"/> Hathor - Default	Administrator	hathor	☆		5

At the bottom of the table, there is a 'Display # 20' dropdown menu.

Figure 8.1 Template Manager in the back end

The term *style* is probably confusing for some people. Style here refers to different versions of the same template. These versions may differ in both CSS styling and HTML markup. The use of styles is further explained in Chapter 11, “The XML File and the Template Parameters.” But let’s get back to the Template Manager.

In addition to viewing the selected styles, you can use the Template Manager to see whether it is a frontend or backend style, which template the selected style relates to, or if it has been assigned to specific page areas. The gold star indicates the selected, and therefore currently active, default style.

By clicking on the checkbox in front of each style and selecting Make Default at the upper right, you can change the default style.

At the top right are also two checkboxes for filtering styles. This function is helpful if you are using many different styles and different templates. The styles can be filtered through the front end or back end or by the template they belong to.

Each template can have different properties, called parameters, that can be configured. By copying a style, you can assign it to different pages or areas with different properties. You can do it both in the Template Manager and via the menu items.

Here is a little example to make things clearer. In the *beez_20* template you can choose between two different design variations: *Personal* and *Nature*. *Personal* is the default display option with the blue header image. *Nature* is in all green. Both variations differ only in that they use a different CSS file for certain elements of the template.

If you want to use the default variation on some pages and the green version on others, you can do so very easily. You can copy the styles from the Template Manager and save them under a different name.

If you then click on the style *beez_20*, you will see the image shown in Figure 8.2.

Tip

Creating multilingual content was not yet possible in Joomla! version 1.5. Now, Joomla! offers a small but smart method of managing content even in different languages. The solution offered is not appropriate for all cases. Sites with a lot of content must fall back on external solutions. But for smaller pages, it is a simple and quick solution.

On the right you can see the parameters, here called Advanced Options. The selection of parameters actually constitutes a style. You have a range of selection options. At the bottom under Template color are the styles *Personal* and *Nature*. Here you can make your selection and choose the desired display variation.

Under Details you will find the name of the relevant style. You can choose any name, but you should make sure it’s a meaningful title to avoid confusion later. The name will help you distinguish between styles, especially if you are using several copies of a style.

You can also see which template the selected style relates to, which unique ID it has, and whether it should be used only when a certain language is selected.

Below the Details panel you will find the Menus assignment panel. Each menu item is listed there, and the style can be assigned to the relevant menu item by enabling

The screenshot shows the Joomla! Template Manager interface for editing the 'Beez2 - Default' style. The interface is divided into several sections:

- Header:** 'Template Manager: Edit Style' with navigation icons for Save, Save & Close, Save as Copy, Close, and Help.
- Details:**
 - Style Name: Beez2 - Default
 - Template: beez_20
 - Site: All (dropdown)
 - Default: All (dropdown)
 - ID: 4
 - Template description: Accessible template for Joomla! Beez, the HTML 4 version.
- Advanced Options:**
 - Wrapper Small (%): 53
 - Wrapper Large (%): 72
 - Logo: images/joomla_black.gif (with Select and Clear buttons)
 - Site Title: Joomla!
 - Site Description: Open Source Content Manai
 - Position of Navigation: before content (dropdown)
 - Template colour: Personal (dropdown)
- Menus assignment:**
 - Menu Selection: Toggle Selection (button)
 - About Joomla:**
 - Getting Started
 - Using Joomla!
 - - Using Extensions
 - - - Components
 - - - - Content Component
 - - - - Single Article
 - - - - Article Categories
 - - - - Article Category Blog
 - - - - Article Category List
 - - - - Featured Articles
 - Australian Parks:**
 - Parks Home
 - Park Blog
 - Write a Blog Post
 - Image Gallery
 - - Animals
 - - Scenery
 - Park Links
 - Fruit Shop:**
 - Welcome

Figure 8.2 Default style beez_20

the corresponding checkbox. By selecting Toggle Selection, you can disable all the checkboxes.

Once you have made your selection, you can save your style by clicking on Save as Copy at the top right. Joomla! automatically creates a copy of the default style with the options you have selected. And that's it!

Here is a summary of all the steps.

1. Select default style.
2. Change title or name of style.
3. Adapt parameters to your preferences.
4. Assign menu items.
5. Save template as copy.

The Template Manager: Templates

If you select the tab Templates in the horizontal navigation, you will get straight to the installed templates, as shown in Figure 8.3. At first glance, this view does not seem very spectacular, but it is clearly a very different picture than in Joomla! version 1.5.

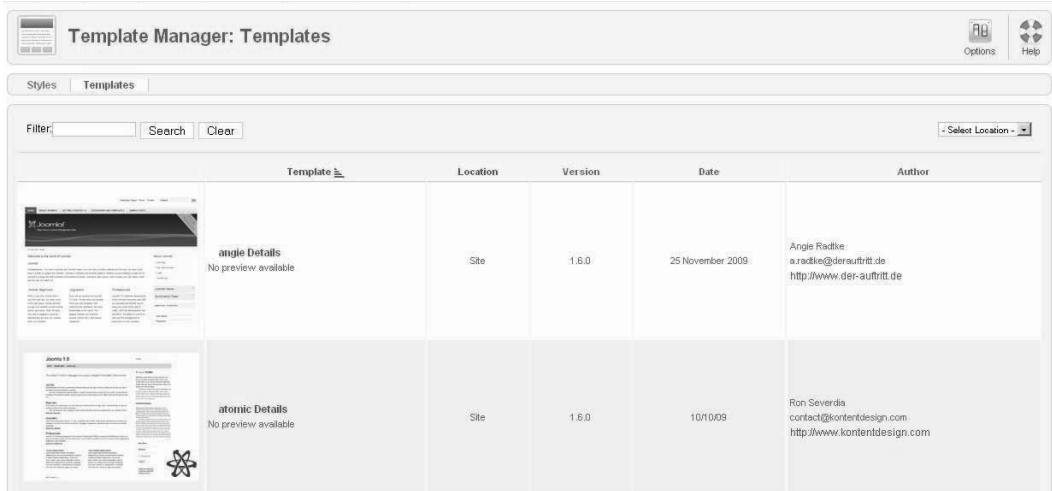


Figure 8.3 Template Manager

The templates are still listed with a screenshot, their name, their location (site for front end or Administrator for back end), their date of creation, and their author. By clicking on the template name, you get to the really interesting information.

The Template Preview

You will notice that under the template name in Figure 8.3 there is a note that says “No preview available.” If you have already been working with Joomla!, you know that you can assign modules, which usually output dynamic contents, to certain positions. In the template itself you control where a module should be placed within the document. Its position is then determined by using CSS. The template preview gives you an overview of the position used for the module.

For security reasons, you have the option of enabling or disabling the preview. You can find the Template Options in the top right corner. When you click on the Options icon, you will see the screen shown in Figure 8.4.

Here you can enable the preview function and configure the permissions management. Once you enable the preview function, you will see, as shown in Figure 8.5, that the preview is now available.

Where in Figure 8.3 it said “No preview available” under the template name, you will now see “Preview,” which is a link that takes you directly to the preview (if Preview does not appear after you change the options, try reloading the page). If you then click on Preview, a new window pops up with the preview. All module positions assigned in the template are displayed clearly.

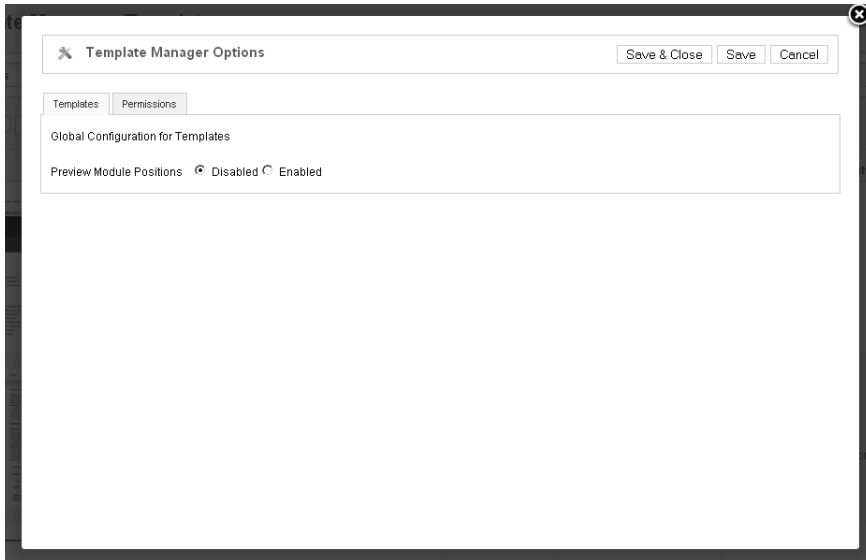


Figure 8.4 Template Manager Options

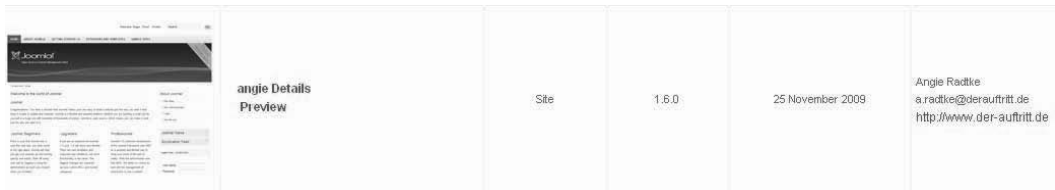


Figure 8.5 The preview is now available.

The latter part of the URL is particularly important: `tp=1&template=beez5`. This helpful function can be used not only in the back end but also in the front end, where it is even more helpful. See Figure 8.6.

Particularly with large sites, you may see a module displayed in the front end but, for the life of you, cannot remember which module position it uses. In the Module Manager, modules can be sorted by type and position, but sometimes you still cannot find a specific module.

If you now go the relevant page in the front end and append `?tp=1` to the URL, you get a preview (of the assigned template) of the corresponding page, which tells you the position of the module you are interested in. This is shown in Figure 8.7.

You may wonder why you can enable and disable this rather useful function. Wouldn't it be great if it was available all the time? The flexibility of the function makes it necessary to have the option of enabling and disabling it, because anyone who knows about

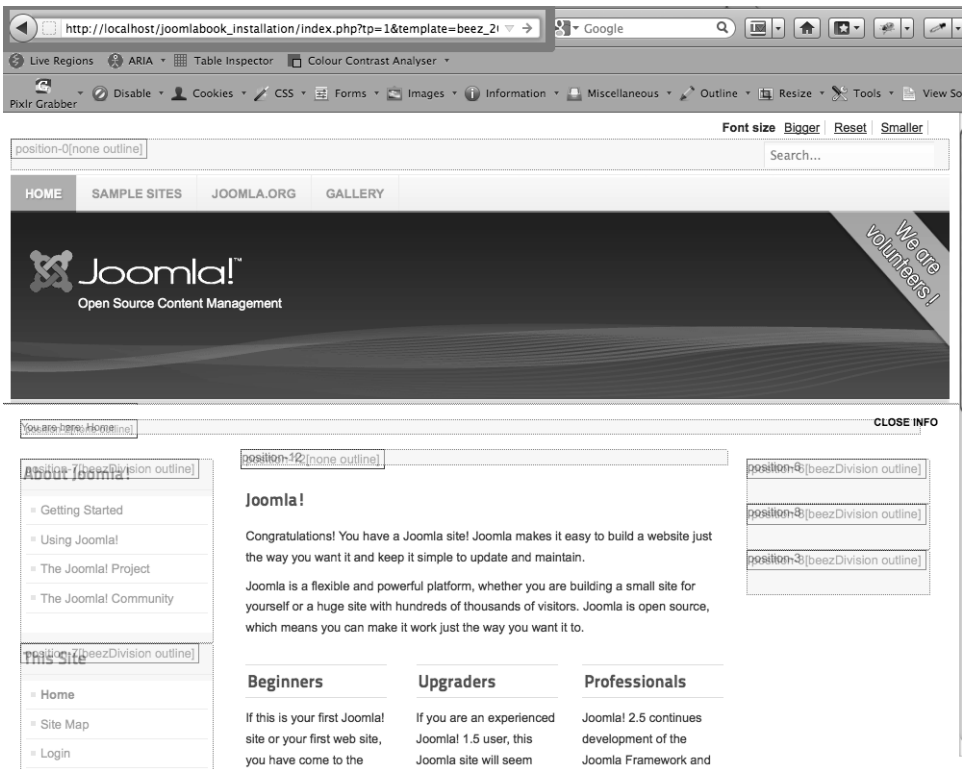


Figure 8.6 Template preview

this method can look up this information. And only rarely would you want them to. So, if you require this function: *enable, look it up, disable!*

Template Details

As you can see in Figure 8.8, the Template Details view also differs slightly from the previous version of Joomla!.

On the right is a list of all CSS files used in the template.

With just one click, you can edit from the administrator back end. The same goes for the internal control files in the template. Here you can edit the `index.php` file, the heart of the template, the error page, and print preview.

This function is useful while running the operation if you want to change something quickly. But I have to admit that I prefer editing these files directly in an editor and then uploading them to the server via FTP.

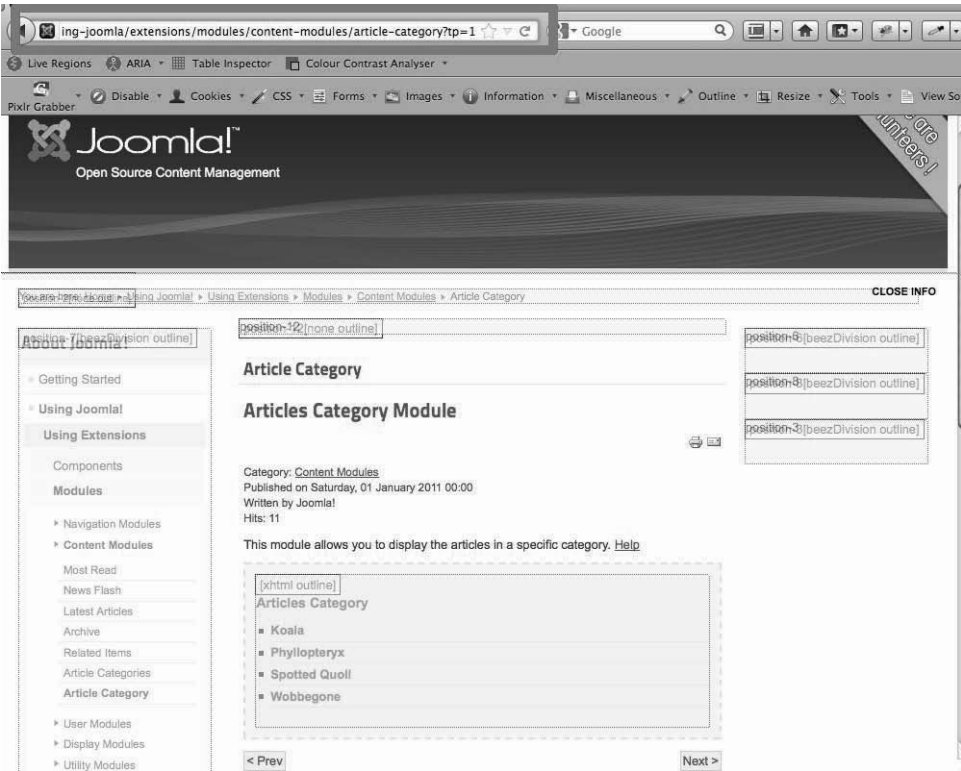


Figure 8.7 Preview in the front end

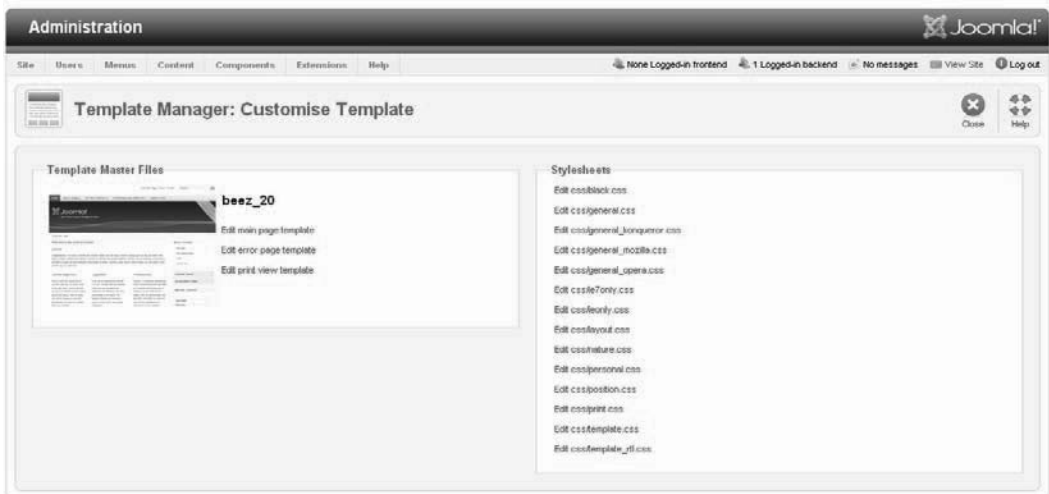


Figure 8.8 Template Details gives access to template page HTML and CSS files.

Installing Templates

When people start getting into Joomla!, they often do not build their own template right away but instead use one of the many templates available on the Internet and adapt it to their needs. Quite a number of templates are now available. In addition to free templates, various templates can be purchased on the Web. The templates differ not just in their design but also in their range of functions, quality, and price.

If you are considering one of these, you should look at the templates very closely. Most of them offer a whole range of functions in addition to the pure design and try to be as generic as possible to fulfill as many wishes as possible. For that reason, these templates are often very hard to adapt, because the more complex they are, the more complicated they are to change.

Now that Joomla! 2.5 has been out for a while, the number of templates offered is extensive.

Templates are managed by Joomla! in the same way as any other extension; that is, they are installed in the same way as any other extension, via the extension manager. Your template is probably in the form of a ZIP archive. You can install this archive, as is, via the extension manager. Joomla! takes care of unzipping the archive. How to create such archives yourself and which rules you need to follow are explained in Chapter 11, “The XML File and the Template Parameters.” So to install a template you go to Extensions → Extension Manager on the Install tab, as shown in Figure 8.9.

If the template is well formed and follows all the Joomla! guidelines, there should not be any problems in the installation, and the template should appear in the Template Manager.

Starting with version 1.6, there has been a considerable change in template handling: they are now stored in the database. In version 1.5 you were still able to simply copy templates into the Joomla! template folder. They were fully functional and automatically detected by the system. This is no longer possible, which does not necessarily mean that it is more complicated. The Extension Manager’s *Discover* function, shown in Figure 8.10, is available to help.

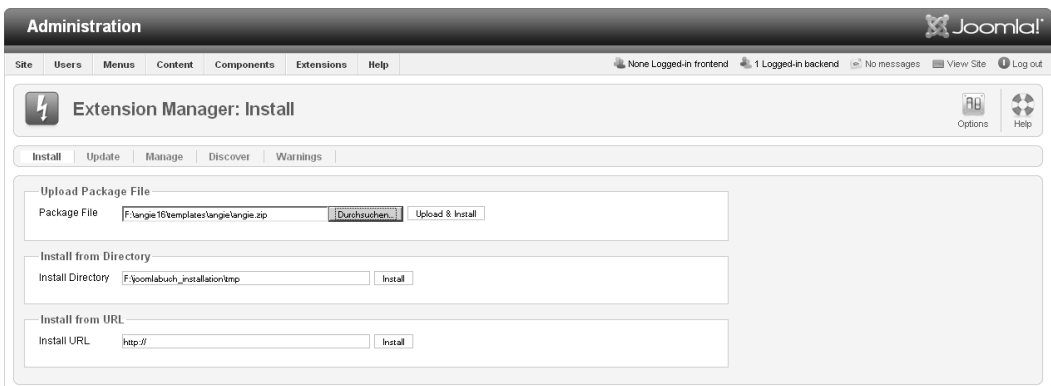


Figure 8.9 Extension manager



Figure 8.10 The Template Manager's Discover function

Once you have manually inserted the template into the Joomla! template directory, go to the Extension Manager on the Discover tab. Click the Discover icon, and the system will detect your template. Check the box in front of your template and click the Install icon. It will then be listed in the Template Manager next to the default templates.

Of course, the question is why the templates are now saved in the database. Templates have, as previously mentioned, parameters: properties that can change under certain circumstances. Like the template name, these properties are also saved in the database, which in turn enables extensions such as modules or components to directly access these properties. This method of storage makes it possible to design a very flexible template.

Index

- * (asterisk)
 - asterisk hack, 52–53
 - wildcard character, 42–43
- : (colon), ending if statements, 67
- , (comma), in CSS element selectors, 40
- . (period), in CSS class selectors, 41
- \$(), dollar sign function, 73–74
- \$ (dollar sign), PHP variable indicator, 66
- \$\$(), double dollar function, 73–74
- < (left angle), less than operator, 67
- <= (left angle, equal sign), less than or equal to operator, 67
- <% ... %>, PHP section delimiters, 64
- <? ... ?>, PHP section delimiters, 64
- <?PHP ... ?>, PHP section delimiters, 64
- “ (double quote), in PHP variables, 66
- = (equal sign), assigning values to PHP variables, 66
- == (equal signs), equal operator, 67
- === (equal signs), identical operator, 67
- != (exclamation, equal sign), not equal operator, 67
- !== (exclamation, equal signs), not identical operator, 67
- # (pound sign), CSS ID indicator, 44
- > (right angle), greater than operator, 67
- >= (right angle, equal sign), greater than or equal to operator, 67
- ' (single quote), in PHP variables, 66
- /*...*/ (slash asterisk), PHP comment delimiters, 65

// (slashes), PHP comment delimiter, 65
{ } (curly brackets), in CSS element selectors, 40
; (semicolon)
 in CSS element selectors, 40
 ending PHP statements, 65
400 – 500 error codes in messages, 178
404 (File not found) messages, 104. See *also* **Error messages; System notices.**

A

Accessibility

Beez template features, 194–196
government sector, 20–21
overview, 19–20
private sector, 21
testing, 257

Accessibility, tools and utilities

CCA (Colour Contrast Analyzer), 86–87
Firebug, 86–90
for Firefox, 87–90
for Internet Explorer, 87–90
Juicy Studio Accessibility extension, 86
rules checkers, 88–89
Wave, 88
WCAG 2 compatibility checker, 89

Accessibility Evaluation Toolbar, 88

Accessibility Overview, 279

Accessibility Toolbar Mozilla/Firefox, 281

Acoustic output, 27

addEvent() method, 73

AIS Web Accessibility toolbar, 87–88

Ajax, MooTools, 76

Animations, 73, 75–76

archive view, com_content component, 293

article view, com_content component, 288–289

Articles

archived, 308
changing number of, 223
headings, 309
latest, 309
listing, 309, 312
most popular, 309
Read More links, 309
separators, 309

<aside> element, 270

Assistive technologies, 279

Asterisk (*)

asterisk hack, 52–53
wildcard character, 42–43

Asymmetrical curvatures, creating, 56

Atomic template

Blueprint framework, 207–209
overview, 91
reference summary, 286–287

Attribute selectors, 44

Audiodata, 27

Author, specifying

com_content component, 288–290, 292–293
mod_articles_category, 308

autocomplete attribute, 130–131

Autocompletion, enabling, 130–131

B

Background images, 249–253, 260

background-color property, 259

background-image property, 259

background-position property, 259

background-repeat property, 259

Banners, 309

Barrier-free design. See **Accessibility.**

BAUM Retec AG, 27

Beez templates

- music store example, 212–216
- overview, 212–216
- styles, 146–148

Beez templates, most important features

- accessibility, 194–196
- custom font sizing, 205
- heading hierarchy, 195–196
- hiding/showing page elements, 199–200
- HTML5, 205–207
- index.php file, 206
- JavaScript, 199
- landmark roles, 199–200
- linearization, 194–196
- positionable navigation column, 197–199
- selectable design, 196
- semantically logical structure, 194–196
- separation of content and layout, 194
- summary of, 193
- tab presentation, 204–205
- WAI-ARIA specification, 199–200

beez5 template

- Chrome Template, 307–308
- overview, 91–92
- reference summary, 284–286

beez_20 template. See also index.php file.

- Chrome Template, 305–306
- overview, 91–92
- reference summary, 283–284
- renaming, 119–123

beezDivision style, 305–306, 307**beezHide style, 147–148, 306, 307****beezTabs style, 147–148, 306, 307–308****Berners Lee, Tim, 35****Blindness. See Visual impairment.****Blindows screen reader, 27****blog view, com_content component, 291–292****Blokland, Eric van, 14****Blooming, 11****Blueprint framework, Atomic template, 207–209****Books and publications. See also Useful links.**

- Bulletproof Web Design*, 36
- Cascading Style Sheets*, 35
- HTML5 & CSS3 Visual QuickStart Guide*, 36
- Pro JavaScript with MooTools*, 81
- Responsive Web Design*, 59
- Sam's Teach Yourself Web Publishing...*, 36
- Transcending CSS: The Fine Art of Web Design*, 5

border-radius statement, 56**Borders, 47–51****Box model, 47–51****box-shadow statement, 56–57****Braille displays, 26–27****Browser check, 243–245, 255–256, 264–265****Browser problems. See also specific browsers.**

- * (asterisk), asterisk hack, 52–53
- conditional comments, 52
- display problems, avoiding, 55
- hasLayout property, 53–54
- vendor prefixes, 55

Browser windows, determining height and width, 60**Browsers**

- detecting, 187–189
- icons in. *See Favicons.*
- passing values from PHP, 65

Bulletproof Web Design*, 36*Buttons, music store example, 228–230**

C

Cascading Style Sheets, 35

Cascading Style Sheets (CSS). *See* **CSS (Cascading Style Sheets).**

Castro, Elizabeth, 36

Categories, listing, 308

categories view

com_contact component, 295

com_content component, 289

com_newsfeeds component, 299

com_weblinks component, 301

category view

com_contact component, 295–297

com_content component, 289–292

com_newsfeeds component, 299–300

com_weblinks component, 301–302

CCA (Colour Contrast Analyzer), 86–87, 281

Cederholm, Dan, 36

Character set, specifying, 110

Charlier, Michael, 7

Checker tools, 281

Child elements, getting, 75

Child selectors, 42

Chrome, Beez styles, 146–148

Chrome Template beez5, 307–308

Chrome Template beez_20, 305–306

Clarke, Andy, 5

class attribute, 130

Class selectors, 41

Class system, MooTools, 76–79

Classes. *See also* **CSS classes.**

JForm, 126–129

moduletable class, 144

Request, 76, 80

Request.HTML, 76, 80

Request.JSON, 76, 80

Class() function, 77

clear property, 51

Colburn, Rafe, 36

Colon (:), ending if statements, 67

Color gradients, 260

Color perception, simulating, 87

Colorblindness

definition, 8

men *vs.* women, 21

in the population, 21

simulating, 8, 10, 87

Colors

background images, 259

blooming, 11

complimentary, finding, 8

contrasts, 11–12

effects of, examples, 9

effects on learning disabilities, 32

influence on accessibility, 12. *See also* Colorblindness.

inverting screen color, 26

linear gradients, 57

links, specifying, 43–44

Colour Contrast Analyzer (CCA), 86–87, 281

Column layout

examples, 214–216

visual structure, 3

Columns

borders, 287

combining, 149–151

empty, hiding, 150, 282

filling with content, 218–222

hiding/showing, 200–202

for low resolution devices, 59–60

number of, determining, 44–47, 287

rows containing, 287

scaling, 16

specifying in percentages, 59–60

three-column design, example, 6

width, fluid page layouts, 16

com_contact component, 294–299

com_content component, 287–294

Comma (,), in CSS element selectors, 40

Comments, PHP, 65

com_newsfeeds component, 299–301

Complimentary colors, finding, 8

component.php file

customizing views, 182–183

replacing system graphics, 179–183

search engine-friendly URLs,
181–182

as template component, 103

Components. *See also* **CSS components**.

dynamic content, 114–115

vs. modules, 114

com_search component, 303

com_users component, 304–305

com_weblinks component, 301–302

com_wrapper component, 305

Conditional comments, browser
problems, 52

Conditional processing, PHP

else statements, 68

if statements, 66–67

relational operators, 67

<config> element. *See also* **Template
parameters**.

file and folder lists, adding, 134

form elements, adding, 129–134

form fields, adding, 126–129

images, adding, 132–133

menu items, creating, 134

overview, 125–126

radio buttons, adding, 132

select boxes, adding, 131–132

spacers, adding, 133–134

text editing, 134

text fields, adding, 129–131

text fields, validating, 130–131

time zone selection box, creating,
134

Consistent layout, user expectations, 5–6

contact view, **com_contact** component,
297–299

Content, separating from layout, 194

Context selectors, 41–42

Contrasts, effects of, 11–12

Core component, MooTools, 73

Corner radii, specifying, 56

Counting modules, 282

countModules() function, 149–151

CSS (Cascading Style Sheets). *See also*
**HTML (HyperText Markup Language);
Templates**.

assigning to specific media types,
59

Cascading Style Sheets, 35

file folder, 102

history of, 4–5, 35–36

*HTML5 & CSS3 Visual QuickStart
Guide*, 36

integrating with JavaScript,
112–113

for low resolution devices, 59–62

vs. module styles, 144

RTL (right-to-left) views, 186–187

Sam's Teach Yourself Web Publishing...,
36

*Transcending CSS: The Fine Art of Web
Design*, 5

useful links, 279–280

validation, 83–84, 85–86, 253

CSS classes

class selectors, 41

grouping, 44–47

CSS classes, reference summary

Atomic template, 286–287

beez5 template, 284–286

beez_20 template, 283–284

components, 287–305. *See also* *specific
components*.

modules, 305–313

CSS components

- com_contact, 294–299
- com_content, 287–294
- com_newsfeeds, 299–301
- com_search, 303
- com_users, 304–305
- com_weblinks, 301–302
- com_wrapper, 305

css folder, 102**CSS selectors**

- attribute, 44
- child, 42
- class, 41
- context, 41–42
- element, 40
- ID, 41
- inheriting properties, 44
- pseudoclass, 43–44
- universal, 42–43

CSS selectors, selective formatting

- direct child of a body, 42
- document elements, 40
- HTML attributes, 44
- individual page areas, 41–42
- link color, 43–44
- with pseudoclasses, 43–44
- selected paragraphs, 41
- with wildcards, 42–43

CSS statements, adding

- in the document head, 39
- in external files, 38–39
- as inline styles, 39–40

CSS statements, most useful

- border-radius, 56
- box-shadow, 56–57
- corner radii, specifying, 56
- linear-gradient, 57
- min-width, 60
- radii of corners, specifying, 56
- shadows, adding to elements, 56–57

CSS3, 55, 60**CSSTidy, CSS tuning tool, 55****Curly brackets ({}), in CSS element selectors, 40****Curvatures, creating, 56****Customizing views, 182–183****Cynthia Says, 281**

D
Dates

- of creation, 288, 289
- current, getting, 282
- formatting, 191–192
- of modification, 288, 289
- outputting, 190
- of publication, 288, 289

Deafness. See Hearing impairment.**Deprecated attributes, 36****Design, 280****Developer Toolbar for Internet Explorer, 281****Devices. See also Media.**

- maximum width, determining, 61
- orientation, determining, 60
- testing on, 61

Direct child of body, 42**Disabilities. See also Hearing impairment;****Learning disabilities; Physical disabilities; Visual impairment.**

- contextual disabilities, 19–20
- effects of aging, 33–34

Discovering templates, 99–100**Display problems, avoiding, 55****Displaying values. See \$this object.****div container, music store example, 236****Document body, index.php file, 114–115****Document head, index.php file, 107–113**

Document properties, loading, 282
Document title
 getting, 282
 specifying, 111
Document type, selecting, 108
document.id() function, 73
Dojo framework, 72
Dollar sign (\$), PHP variable indicator, 66
Dollar sign function, \$(), 73–74
Double dollar function, \$\$(), 73–74
Double quote (“), in PHP variables, 66
Drop-down menus, 157–158
Dynamic content. See also Modules; MooTools.

error messages, 114–115
 Joomla! components, 114–115
 modules, 114–115
 system notices, 114–115

Dynamic screen design. See Responsive Web design.

E

echo() function, 65
Elderly people, disabilities of. See Users, age and accessibility.
Element selectors, 40
else statements, 68
Equal signs
 =, assigning values to PHP variables, 66
 ==, equal operator, 67
 ===, identical operator, 67
Error messages. See also System notices.
 400 - 500 error codes, 178
 404 (File not found), 104
 beez5 template, 286
 beez_20 template, 284
 dynamic content, 114–115

incorrect password, 114–115
 “The template for this display is not available.”, 119–120
error.php file, 104, 273
Events, 73–74
Examples. See Music store example.
Exclamation, equal sign
 !=, not equal operator, 67
 !=, not identical operator, 67
Extends property, 77
Eye-tracking experiments, 6

F

Faulkner, Nils, 86
favicon.ico file, 104
Favicons, changing, 112, 275
featured view
 com_contact component, 294–295
 com_content component, 287–288
Fields. See Form fields.
File lists, adding, 134
Files, removing superfluous, 274
filter attribute, 130
Firebug, 85–90, 281
Firefox
 Accessibility Evaluation Toolbar, 88
 custom style sheets for visual impairment, 23–24
 Mozilla Accessibility toolbar, 87–88
 Tilt 3D extension, 89–90
 Web Developer toolbar, 84–85
 Web page inspection tool, 89–90
Fixed page layouts, 16
float property, 49
Floated elements, 49–51
Fluid page layouts, 16
Folded out menus, 157–158
Folder lists, adding, 134

Fonts. See *also* **Text**.

- browser support for, 14–15
- custom sizing, 205
- designing for the visually impaired, 29
- enlarging, 241–242
- Google Font API, 15–16
- serif *vs.* sans serif, 13
- size. See *Scaling*, fonts.
- typography, 280–281
- Web, 13–16
- WOFF (Web Open Format Font), 14–15

fonts folder, 104–105**<footer> element, 270****Footers**

- adjusting, 239–240
- copyright information, 310
- copyright notes, 287
- enclosing, 283, 284
- <footer> element, 270
- license information, 310
- specifying, 284, 286

Form elements, adding, 129–134**Form fields, adding, 126–129****form view**

- com_content component, 293–294
- com_weblinks component, 302

Formatting selected areas, 151–155. See *also* **Classes**; **CSS selectors**, *selective formatting*.**400 - 500 error codes in messages, 178****404 (File not found) messages, 104****Freedom Scientific, 27****Functions**

- \$(), dollar sign, 73–74
- \$\$(), double dollar, 73–74
- Class(), 77
- countModules(), 149–151
- document.id(), 73
- echo(), 65

- getChildren(), 75
- getNext(), 75
- getParent(), 75
- modChrome, 202
- MooTools, 73–74
- print_r(), 69
- reference summary, 282
- var_dump(), 69

Fx.Morph object, 75–76**Fx.Tween object, 75–76**

G

getChildren() function, 75**getNext() function, 75****getParent() function, 75****Getting values.** See **\$this object**.**Golden ratio, 3, 16****Google Font API, 15–16****Graphical structure.** See **Page layout**, **graphical structure**.**Graphics.** See *also* **Images**.

- optimizing for specific devices, 61–62
- system, replacing, 179–183

Greater than operator (>), 67**Greater than or equal to operator (>=), 67****Grids, 3–4****GUI (graphical user interface).** See **Chrome**.

H

hasLayout property, browser problems, 53–54**Header area**

- bee5 template, 284
- bee5_20 template, 283

<header> element, 269**Header pictures, module position, 235–239**

Headers

- background images, 259–265
- <header> element, 269
- transparent, 29
- visual design, 223–234

Heading hierarchies

- Beez template features, 195–196
- modules, 146–148

Hearing impairment

- effects of aging, 34
- overview, 31
- sign language videos, 31

Height

- browser windows, 60
- content, minimum, 240–241
- content, music store example, 240–241
- line-height property, 13
- media, 60

Hemberger, Frederic, 61**hide.js file, 201–202****Hiding/showing elements. See also Invisible.**

- Beez template features, 199–200
- beezHide style for, 306, 307
- collapsing content, 306, 307
- columns, 200–202, 284, 286
- elements outside the viewport, 283, 285
- modules, 148, 202–204
- page elements, 199–200
- parameter return values, 70

Hit counter, 288, 289**Holzschlag, Molly, 5****Homepage structuring, 190****HTML (HyperText Markup Language). See also**

- CSS (Cascading Style Sheets).** choosing a version, 36–38. *See also specific versions.*
- codes for special characters, 128

history of, 35–36

HTML5 & CSS3 Visual QuickStart Guide, 36

mixing with PHP, 63–64

Sam's Teach Yourself Web Publishing..., 36

validation, 83–84, 85–86, 253–254

HTML 4.01

deprecated attributes, 36

overview, 36–37

Strict version, 36–37

Transitional version, 36–37

html folder, 103**HTML language indicator, 108–109****HTML5**

<aside> element, 270

Beez template features, 205–207

<footer> element, 270

<header> element, 269

index.php file, 269

<nav> element, 270

outputting content as, 205–207

overriding, 267–269

overview, 37–38

sample of sites using, 37

using effectively, 265–271

vs. XHTML, 266

HTML5 & CSS3 Visual QuickStart Guide, 36

I

Icon list, 288**Icons**

- edit, 288
- e-mail, 288
- favicons, 104
- print, 288
- useful links, 281

ID selectors, 41

Identical operator (===), 67

if statements, 66–67

Images. *See also* **Graphics.**

- adding, 132–133
- background, accessibility, 260
- background, examples, 249–253
- color gradients, 260
- random, 312

images folder, 103

implement() method, 77

index.html file, 105

index.php file

- accessing views, 190
- Beez template features, 206
- character set, specifying, 110
- CSS, integrating with JavaScript, 112–113
- current date, formatting, 191–192
- current date, outputting, 190
- definition, 102
- describing site contents, 110–111
- document title, specifying, 111
- document type, selecting, 108
- favicons, specifying, 112
- HTML language indicator, 108–109
- HTML5, 269
- integrating modules into, 139–148
- jdoc: include type:head statement, 109
- language files, 137–138
- metadata for search engines, specifying, 110–111
- MooTools, integrating, 113
- music store example, 221–222
- optimizing, 275–277
- reading direction, changing, 113
- restricting access to, 107–108
- RSS feeds, enabling, 111–112
- search engine-friendly URLs, generating, 109

- stopping robots, 110–111
- structuring the homepage, 190
- UTF-8 character set, specifying, 110

index.php file, structure of

- document body, 114–115
- document head, 107–113

Inheritance, CSS selectors, 44

Installing

- Joomla!, 123
- templates, 99–100

Internet Explorer

- * (asterisk), asterisk hack, 52–53
- AIS Web Accessibility toolbar, 87–88
- browser check, 243, 264–265
- conditional comments, 52
- CSS child selectors, 42
- custom style sheets for visual impairment, 23–24
- hasLayout property, 53–54
- HTML5, 37–38, 271
- max-width property, 17
- optimizing for visual impairment, 23
- rounding corners, 145–146

Invisible elements, 29. *See also* **Hiding/showing.**

iPad, customizing views for, 182–183

iPhone, customizing views for, 182–183

J

JavaScript

- Beez template features, 199
- experimentation tools, 81
- integrating with CSS, 112–113
- useful links, 281

javascript folder, 103

JavaScript frameworks, 72. *See also* **MooTools.**

JAWS (Job Access with Speech), 27, 279

jdoc: include type:head statement, 109

jquery:include, 139–148
JForm class, 126–129
Joomla!, 279
jQuery framework, 72
JSFiddle, 81
Juicy Studio Accessibility extension, 86

K

Kew, Jonathan, 14
Keyboards
 navigation problems, 230–234
 onscreen, 25–26
 testing, 242
Keyword spamming, 110
Kröner, Peter, 72

L

Landmark roles, 199–200
Language file
 folder for, 103, 137–138
 location, 136
 music store example, 222, 234
 path to, specifying, 124
 renaming, 124
 rules for, 143
language folder, 103
Language strings
 conventions for using, 137
 music store example, 234
Languages. See also RTL (right-to-left)
languages.
 adding, 136
 changing, 310
 file folder, 103
 getting, 282
 multilingual content, 93
 reading direction, changing, 113
 RTL (right-to-left) reading, 113

 simplifying for learning disabilities, 33
 translating constants, 135–136
Layout. See Page layout.
Learning disabilities
 accommodating, 32–33
 colors, effects of, 32
 overview, 32
 simplifying language, 33
Left angle, equal sign (<=), less than or equal to operator, 67
Left angle (<), less than operator, 67
Legal definition of blindness, 22
Legal mandate for accessibility, 20–21
Lemay, Laura, 36
Leming, Tal, 14
Less than operator (<), 67
Less than or equal to operator (<=), 67
Line spacing, 13
Linear gradients, 57
linear-gradient, 57
Linearization, 194–196
line-height property, 13
Links. See also specific links.
 breadcrumbs, 310
 color, specifying, 43–44
 images as, 159–160
 nonvisual. *See* Skip links.
 to open and close, 306, 307
 push-to-front principle, 7
 text, 284, 286
 titles, 160
Login form, 311
login view, com_users component, 304
Logos
 alternative text, 232–233
 beez5 template, 285
 beez_20 template, 283
 formatting, 225–228
 template parameters, 124
Logout button, 311

M

Mac OS X accessibility aids, useful links, 27

Magnifying

- fonts. *See* Scaling, fonts.
- screen magnifiers, 24–25

Marcotte, Ethan, 59

margin property, 49

Margins, 47–51

max-device-width property, 61

max-width property, 17

Media. *See also* Devices.

- automatically adapting to, 59–62
- height and width, determining, 60
- querying, 59–62

Media Manager, selecting images from, 132–133

Media queries

- CSS3, 60–61
- HTML5 applications, 62

Menu items

- creating, 134
- wrapper type, 183

Menus

- displaying, 155
- drop-down, 157–158
- folded out, 157–158
- horizontal navigation, 156–157
- invisible, 181–182
- music store example, 223–225
- navigation, positioning, 218
- split, 158
- styling, 159
- subnavigation, 156–157

Messages, types of, 173. *See also* **Error messages; System notices.**

Metadata for search engines, specifying, 110–111

Methods. *See also* **Functions.**

- addEvent(), 73
- displaying. *See* \$this object.

implement(), 77

tween(), 73

Meyer, Eric, 35

min-width statement, 60

mod_articles_archive module, 308

mod_articles_categories module, 308

mod_articles_category module, 308

mod_articles_latest module, 309

mod_articles_news module, 309

mod_articles_popular module, 309

mod_banners module, 309

mod_breadcrumbs module, 310

modChrome functions module, 202

mod_custom module, 310

Model-view-controller principle, 166–172

mod_feed module, 310

mod_footer module, 310

mod_languages module, 310

mod_login module, 311

mod_menu module, 311–312

mod_random_image module, 312

mod_related_items module, 312

mod_search module, 312

mod_stats module, 312

mod_syndicate module, 313

Module Class Suffix

- formatting selected page areas, 151–155
- music store example, 236–239

Module contents, beezDivision style for, 305–306, 307

Modules

- arranging in tabs, 147–148
- boxes for, 284, 286
- Chrome Template beez5, 307–308
- Chrome Template beez_20, 305–306
- vs.* components, 114
- counting, 282
- dynamic content, 114–115
- editing content, 261

- formatting selected page areas, 151–155
- heading hierarchy, 146–148
- hiding/showing, 148, 202–204
- integrating into index.php file, 139–148
- link images, 159–160
- link titles, 160
- mod_articles_archive, 308
- mod_articles_categories, 308
- mod_articles_category, 308
- mod_articles_latest, 309
- mod_articles_news, 309
- mod_articles_popular, 309
- mod_banners, 309
- mod_breadcrumbs, 310
- mod_custom, 310
- mod_feed, 310
- mod_footer, 310
- mod_languages, 310
- mod_login, 311
- mod_menu, 311–312
- mod_random_image, 312
- mod_related_items, 312
- mod_search, 312
- mod_stats, 312
- mod_syndicate, 313
- mod_users_latest, 313
- mod_weblinks, 313
- mod_whosonline, 313
- mod_wrapper, 313
- most widely used, 139
- naming, 140–144
 - for page layout, 149–151
 - positioning, 284, 286
 - positions, defining, 123–124

Modules, for menus

- displaying menus, 155
- drop-down menus, 157–158
- folded out menus, 157–158

- horizontal navigation, 156–157
- split menus, 158
- styling individual menus, 159
- subnavigation, 156–157

Modules, positioning

- with Module Manager, 143
- by name, 140–144

Modules, styles

- Beez template, 146–148
- beezHide, 147–148
- beezTabs, 147–148
- Chrome, 146–148
- vs.* CSS, 144
- default, 144–146
- for menus, 159

moduletable class, 144

mod_users_latest, 313

mod_weblinks, 313

mod_whosonline, 312–313

mod_wrapper, 312–313

moo.fx plug-in, 72

MooTools

- Ajax block, 76
- animations, 73, 75–76
- child elements, getting, 75
- class system, 76–79
- Core component, 73
- creating and manipulating elements, 74–75
- documentation, 73, 81
- events, 73–74
- functions, 73–74
- integrating, 113
- More component, 73–74
- overview, 72
- parent elements, getting, 75
- principle of, 79–81
- Pro JavaScript with MooTools*, 81
- selecting elements, 73–74
- sibling elements, getting, 75

MooTools, *continued*

- transferring function between projects, 79–81
- useful links, 81
- XML HTTP requests, 76

More component, MooTools, 73–74**Motor disabilities, 30****Mozilla Accessibility toolbar, 87–88****Mozilla Firefox. See Firefox.****Multilingual content, 93****Music store example**

- adding a div container, 236
- background images, 249–253
- with Beez templates, 212–216
- buttons, 228–230
- column layouts, 214–216
- creating a template, 211–216
- customizing typography, 245–246
- filling columns with content, 218–222
- footers, adjusting, 239–240
- header pictures, module position, 235–239
- headers, visual design, 223–234
- index.php file adaptations, 221–222
- keyboard navigation problems, 230–234
- language file adaptation, 222, 234
- language string adaptation, 234
- logo, alternative text, 232–233
- logo area, formatting, 225–228
- main menu adaptations, 223–225
- menu navigation, positioning, 218
- minimum content height, 240–241
- Module Class Suffix, 236–239
- module headings, formatting, 246–249
- navigation, positioning, 218
- number of articles, changing, 223
- order of content, 233–234

- Read More link, 252–253
- renaming module positions, 219–221
- semantics check, 230
- site description, 231–232
- spacing adjustment, 238–239
- templateDetails.xml file adaptations, 222

Music store example, testing

- accessibility checks, 257
- browser check, 243–245, 255–256
- CSS validation, 253
- font enlargement, 241–242
- HTML validation, 253–254
- keyboard operation, 242

N

name attribute, 140–144**Naming**

- beez_20 template, 119–123
- language files, 124
- module positions, 219–221
- modules, 140–144

Nature design, 93**<nav> element, 270****Navigation**

- menus, 156–157
- <nav> element, 270
- positioning, 218
- separating from content, 12

Navigation columns, positioning, 197–199, 283, 285**newsfeed view, com_newsfeeds component, 300–301****Newsfeeds, listing, 310****Newton, Aaron, 81****nofollow keyword, 110****noindex keyword, 110****Not equal operator (!=), 67**

Not identical operator (!=), 67

Notices. See **System notices.**

O

Obcena, Mark Joseph, 81

Object values, displaying. See **\$this object.**

Offline mode, 183

offline.php file, 183, 273

Online resources. See **Useful links.**

Onscreen keyboards, 25–26

Operability, 20–21

Ordering content, 233–234

Orientation of a device, determining, 59–62

Output

default, 161

as HTML5, 205–207

model-view-controller principle,
166–172

overriding templates, 165–166,
170–172

Page Class Suffix, 162–165

shifting to templates, 168–169

Output, adapting

overview, 169–170

system notices, 173–178

Overriding

HTML5, 267–269

templates, 165–166, 170–172

P

padding property, 48

Padding screen elements, 47–51

Page Class Suffix, 162–165

Page Heading parameter, 66–67

Page layout. See also **Components;**

Fonts; Modules; Text.

column width, 16

fixed *vs.* fluid, 16

golden ratio, 16

scaling text, columns, etc., 16

separating from content, 194

Page layout, graphical structure. See also

Colors.

esthetics, 12–13

line spacing, 13

navigation, separating from content,
12

spatial arrangement, 7–8

Page layout, visual structure

column layout, 3

golden ratio, 3

grid patterns, library of, 4

grids, 3–4

implementing, 5–7

push-to-front principle, 7

table layout, 3

three-column design, example, 6

visual cues to more information, 7.

See also **Links.**

Page numbers, specifying, 288

Page titles

bee5 template, 285

bee5_20 template, 283

Pages. See **Web pages.**

Pagetitle parameter. See **Page Heading parameter.**

Papst, Eva, 27–28

Parameters. See also **Template parameters.**

PHP, 68–70

templates, 93

Parent elements, getting, 75

Parse errors, PHP, 65

Password input field, 311

Passwords, error messages, 114–115

Perceivability, 20

Period (.), in CSS class selectors, 41

Personal design, 93

PHP

- accessing objects, 68–70
- comments, 65
- echo() function, 65
- mixing with HTML, 63–64
- output, 65–66
- overview, 63
- parameters, 68–70
- parse errors, 65
- passing values to the browser, 65
- variables, displaying, 69

PHP, conditional processing

- else statements, 68
- if statements, 66–67
- relational operators, 67

Physical disabilities, 30**Pictures.** *See* **Images.****Positioning modules**

- with Module Manager, 143
- by name, 140–144

Positioning screen elements

- borders, 47–51
- box model, 47–51
- floated elements, 49–51
- margins, 47–51
- space padding, 47–51
- useful links, 51
- width, 47–51

Pound sign (#), CSS ID indicator, 44**Previewing**

- print, 273
- sizing previews, 274–275
- templates, 94–97, 104

Print preview, 273**Print stylesheets, 273****print_r() function, 69****Pro JavaScript with MooTools, 81****profile view, com_users component, 304****Proiette, Valerio, 72****Properties**

- background-color, 259
- background-image, 259
- background-position, 259
- background-repeat, 259
- clear, 51
- displaying. *See* \$this object.
- Extends, 77
- float, 49
- hasLayout, 53–54
- line-height, 13
- margin, 49
- max-device-width, 61
- max-width, 17
- min-width, 60
- padding, 48

Prototype framework, 72**Pseudoclass selectors, 43–44****Push-to-front principle, 7**

R
Radii of corners, specifying, 56**Radio buttons, adding, 132****Read More links, 288, 308–309****Reading direction.** *See also* **RTL (right-to-left) languages.**

- changing, 113, 185–187
- displaying, 282

registration view, com_users component, 304–305**Rehabilitation Act Amendment, 20–21****Relational operators, 67****remind view, com_users component, 305****Renaming.** *See* **Naming.****Request class, 76, 80****Request.HTML class, 76, 80****Request.JSON class, 76, 80****required attribute, 130****reset view, com_users component, 305**

Resizing screen elements dynamically.
 See MooTools; Responsive Web design.

Resolution. See Screen resolution.

Responsive Web design, 59–62

Responsive Web Design, 59

Right angle, equal sign (\geq), greater than or equal to operator, 67

Right angle ($>$), greater than operator, 67

Robots, stopping, 110–111

Robustness, 21

Rounding corners, 144–146

Rows, separators, 287

RSS feeds, enabling, 111–112

RTL (right-to-left) languages. See also Languages; Reading direction.
 changing direction, 185–187, 274
 planning proportions for, 113

S

Sam's Teach Yourself Web Publishing..., 36

Sans serif fonts vs. serif, 13

Scaling
 columns, 16
 text, 16

Scaling, fonts
 accommodating visual impairment, 22–23
 custom sizing, 205
 small *vs.* large, 16

Screen magnifiers, 24–25

Screen readers, 26–28

Screen resolution
 adjusting for visual impairment, 22
 designing for low resolution devices, 59–62
 detecting and adapting to, 59–62

Screen width
 minimum, specifying, 60
 specifying in percentages, 59–60

Search engine-friendly URLs, 109, 181–182

Search feature
 creating, 312
 with filters, 290

search view, 303

Security
 restricting access to index.php file, 107–108
 restricting folder access, 105

Security checks, 282

Select boxes, adding, 131–132

Selectable design features, 196

Selecting elements with MooTools, 73–74

Selectors. See CSS selectors.

Semantically logical structure, 194–196

Semantics check, 230

Semicolon (;)
 in CSS element selectors, 40
 ending PHP statements, 65

Separators
 articles, 309
 items, 288
 rows, 287

Serif fonts vs. sans serif, 13

Shadows, adding to elements, 56–57

Sharing your Web sites with others, 183

Showing/hiding. See Hiding/showing.

Sibling elements, getting, 75

Sidebars, 270

Sign language videos, 31

Single quote (‘), in PHP variables, 66

Site descriptions
 beez5 template, 285
 beez_20 template, 283
 music store example, 231–232

size attribute, 130

Skip links
 anchors, 284, 286
 beez5 template, 285

Skip links, *continued*

beeZ_20 template, 283

example, 137–138

Slash asterisk (/...*/), PHP comment delimiters, 65**Slashes (//), PHP comment delimiter, 65****Space padding, 47–51****Spacers, adding, 133–134****Spacing adjustment**

music store example, 238–239

print stylesheets, 273

Spatial arrangement, page layout, 7–8**Special characters, converting to HTML codes, 128****Split menus, 158****Strict version, 36–37****Strings, PHP output, 65****Structure of Web sites, importance of, 1–2. *See also* Page layout.****style attribute, 144–146****Style sheets, customizing for visual impairment, 23–24****Styles**

beeZDivision, 305–306, 307

beeZHide, 147–148, 306, 307

beeZTabs, 147–148, 306, 307–308

inline, avoiding, 39–40

menus, 159

modules *vs.* CSS, 144

templates, 92–94

Subnavigation menus, 156–157**System graphics, replacing, 179–183****System notices. *See also* Error messages.**

adapting, 173–178

beeZ5 template, 286

beeZ_20 template, 284

dynamic content, 114–115

modifying, 173–178

T**Table layout, visual structure, 3****Tabs**

arranging modules in, 147–148

BeeZ template features, 204–205

beeZTabs style for, 306, 307–308

Telecommunications Act, 21**Template Manager**

previewing templates, 94–97

Styles tab, 92–94

template details, 97–98

Templates tab, 94–98

Template parameters. *See also* <config> element.

accessing, 282

for logos, 124

optional template width, 124

uses for, 124–126

templateDetails.xml file

adapting, 277–278

customizing template names,
119–123

definition, 103

integrating files and folders, 123

language files, 123

music store example, 222

overview, 117–119

template_preview.png file, 104**Templates. *See also* Atomic template; beeZ5 template; beeZ_20 template; Music store example.**

backend, location, 101

control files, 102, 104. *See also* index.
php file.

creating, 211–216

current, displaying. *See* Template
Manager.

detail view, 97–98

discovering, 99–100

displaying as thumbnails, 104

- fonts folder, 104
- frontend, location, 101
- installing, 99–100
- names, customizing, 119–123
- Nature design, 93
- overriding, 165–166, 170–172
- parameters, 93
- Personal design, 93
- previewing, 94–97, 104
- shifting output to, 168–169
- width, specifying, 124

Templates, components

- component.php file, 103
- css folder, 102
- error.php file, 104
- favicon.ico file, 104
- fonts folder, 105
- html folder, 103
- images folder, 103
- index.html file, 105
- index.php file, 102
- javascript folder, 103
- language folder, 103
- overview, 102
- table of contents for, 103. *See also*
 - templateDetails.xml file.
- template control files, 102, 104. *See also*
 - index.php file.
- templateDetails.xml, 103
- template_preview.png file, 104
- template_thumbnail.png file, 104

template_thumbnail.png file, 104, 274–275

Text. *See also* Fonts.

- editing, in text boxes, 134
- scaling, 16

Text fields

- adding, 129–131
- validating, 130–131

“The template for this display is not available.”, 119–120

\$this object, 69

Thumbnails, displaying templates as, 104, 274–275

Tiling images, 260

Tilt 3D extension, 89–90

Time zone selection box, creating, 134

Titles. *See* Page titles.

Tools and utilities

- CSS validator, 83–84
- Firebug, 85–86
- HTML validator, 83–84
- JavaScript frameworks. *See*
 - MooTools.
- Tilt 3D, 89–90
- Web Developer toolbar, 84–85

Tools and utilities, accessibility

- CCA (Colour Contrast Analyzer), 86–87
- Firebug, 86–90
- for Firefox, 87–90
- for Internet Explorer, 87–90
- Juicy Studio Accessibility extension, 86
- rules checkers, 88–89
- Wave, 88
- WCAG 2 compatibility checker, 89

Transcending CSS: The Fine Art of Web Design, 5

Transitional version, 36–37

Translating constants to other languages, 135–136

Transparent elements, 29. *See also* Hiding/showing elements.

tween() method, 73

Typography, 280–281. *See also* Fonts.

U

Understandability, 21

Universal selectors, 42–43

U.S. Legal Activities on Web Accessibility, 21**Usability, eye-tracking experiments, 6****Useful links. See also Books and publications.**

accessibility, private sector, 21
 Accessibility Overview, 279
 Accessibility Toolbar Mozilla/Firefox, 281
 assistive technologies, 279
 checker tools, 281
 color, 281
 colorblindness, 10
 Colour Contrast Analyzer, 281
 CSS, 279–280
 CSSTidy, CSS tuning tool, 55
 Cynthia Says, 281
 design, 280
 Developer Toolbar for Internet Explorer, 281
 device testing, 61
 Firebug, 281
 HTML5, 37, 280
 icons, 281
 JavaScript, 81, 281
 JAWS (Job Access with Speech), 279
 Joomla!, 279
 JSFiddle, 81
 linear gradients, 57
 Mac OS X accessibility aids, 27
 MooTools, 81
 responsive Web design, examples of, 59
 typography, 280–281
 Validator, 281
 Virgo, 279
 Vischeck, 281
 VoiceOver, 27
 WAI-ARIA, 281
 Wave, 281

Wcag2-Checker, 281
 Webdeveloper Toolbar, 281
 Windows eyes, 279

User expectations

goals and motivations, 2–3
 layout consistency, 5–6
 recognizing, 2–3

User name, input field, 311**Users**

age and accessibility, 33–34
 latest, listing, 313
 online, listing, 313

UTF-8 character set, specifying, 110

V
Validation

CSS, 253
 HTML, 83–84, 85–86, 253–254
 text fields, 130–131

Validator, 281**van Blokland, Eric, 14****var_dump() function, 69****Variables, PHP**

displaying, 69
 outputting, 65–66

Vendor prefixes, browser problems, 55**Videos, optimizing for specific devices, 61–62****Views**

accessing, 190
 archive, com_content component, 293
 article, com_content component, 288–289
 blog, com_content component, 291–292
 contact, com_contact component, 297–299
 current, getting, 282

- customizing, 182–183
- definition, 69
- integrating from other sites, 183
- iPad, customizing for, 182–183
- iPhone, customizing for, 182–183
- login view, com_users component, 304
- model-view-controller principle, 166–172
- newsfeed, com_newsfeeds component, 300–301
- profile, com_users component, 304
- registration, com_users component, 304–305
- remind, com_users component, 305
- reset, com_users component, 305
- search, 303
- wrapper, 305

Views, categories

- com_contact component, 295
- com_content component, 289
- com_newsfeeds component, 299
- com_weblinks component, 301

Views, category

- com_contact component, 295–297
- com_content component, 289–292
- com_newsfeeds component, 299–300
- com_weblinks component, 301–302

Views, featured

- com_contact component, 294–295
- com_content component, 287–288

Views, form

- com_content component, 293–294
- com_weblinks component, 302

Virgo screen reader, 27, 279

Vischeck, 281

Visual cues to more information, 7. See also **Links.**

Visual impairment. See also **Colorblindness.**

- accommodating, 29
- acoustic output, 27
- adjusting screen resolution, 22
- braille displays, 26–27
- color perception, simulating, 87
- custom style sheets, 23–24
- effects of aging, 34
- inverting screen color, 26
- legal definition of blindness, 22
- onscreen keyboards, 25–26
- outputting screens on nonvisual devices, 27
- overview, 21–22
- scaling font size, 22–23
- screen magnifiers, 24–25
- screen readers, 26–28
- separating content from layout, 28–29

Visual structure. See **Page layout, visual structure.**

VoiceOver, useful links, 27

W

W3C, accessibility recommendations, 21

WAI-ARIA (Web Accessibility Initiative–Accessible Rich Internet Applications), 199–200

Wave, 88, 281

WCAG 2

- compatibility checker, 89
- guidelines for accessibility, 20–21

WCAG1.0 (Web Content Accessibility Guidelines 1.0), 20–21

Wcag2-Checker, 281

Web Developer toolbar, 84–85

Web fonts, 13–16. See also **Fonts; WOFF (Web Open Format Font).**

Web links. See **Links.**

Web Open Format Font (WOFF), 14–15

Web page inspection tool, 89–90

Web pages. *See also* Page.

- basic structure elements, 205
- switching to offline mode, 183

Web sites

- describing site contents, 110–111
- sharing yours with others, 183

Web sites of interest. *See* Useful links.

Webdeveloper Toolbar, 281

WebFormator plug-in, 27

Width

- browser windows, 60
- columns, fluid page layouts, 16
- devices, determining, 61
- Internet Explorer considerations, 17
- max-device-width property, 61
- max-width property, 17
- media, determining, 60
- min-width property, 59–60
- positioning screen elements, 47–51
- screen, 59–60
- screen elements, 47–51
- specifying with template parameters, 124
- windows, specifying, 60

Wildcards, selective formatting with, 42–43

Windows, specifying minimum width, 60

Windows eyes, 279

WOFF (Web Open Format Font), 14–15

Wrapper component, 305

Wrapper type menu items, 183

wrapper view, 305

X

XHTML 1.01

- deprecated attributes, 36
- overview, 36–37
- Strict version, 36–37
- Transitional version, 36–37

XHTML vs. HTML5, 266

XML file. *See* `templateDetails.xml` file.

Y

YUI framework, 72

Z

Zip archive, creating, 278