

APPENDIX



Licenses

This appendix gives the full text of each license that applies to the content in this book. The GNU Free Documentation License applies to all content. The Pylons License applies to the examples that are based on ones in the Pylons documentation. The YUI license applies to the YUI source files you will include as part of the SimpleSite example code and YUI documentation quoted in Chapter 15.

GNU Free Documentation License

sourcecode:: text

GNU Free Documentation License

Version 1.2, November 2002

Copyright (C) 2000, 2001, 2002 Free Software Foundation, Inc.

51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document “free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The “Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “you”. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A “Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A “Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A.** Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B.** List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C.** State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D.** Preserve all the copyright notices of the Document.
- E.** Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F.** Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G.** Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H.** Include an unaltered copy of this License.
- I.** Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J.** Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K.** For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L.** Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M.** Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N.** Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O.** Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled “Endorsements”, provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled “History” in the various original documents, forming one section Entitled “History”; likewise combine any sections Entitled “Acknowledgements”, and any sections Entitled “Dedications”. You must delete all sections Entitled “Endorsements”.

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright (c) YEAR YOUR NAME.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled “GNU Free Documentation License”.

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with...Texts.” line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Pylons License

Note: This license applies to Pylons itself, not to its dependencies. Please check the licenses of the dependencies separately.

Copyright (c) 2005-2008 Ben Bangert, James Gardner, Philip Jenvey and contributors.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of the author or contributors may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING

IN ANYWAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

ALL TEMPLATES GENERATED ARE COVERED UNDER THE FOLLOWING LICENSE:

Copyright (c) 2005-2008 Ben Bangert, James Gardner, Philip Jenvey and contributors.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following condition is met:

The name of the author or contributors may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANYWAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

YUI License

Software License Agreement (BSD License)

Copyright (c) 2008, Yahoo! Inc.

All rights reserved.

Redistribution and use of this software in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of Yahoo! Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission of Yahoo! Inc.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR

SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Sources of Intellectual Property Included in the YUI Library

YUI is issued by Yahoo! under the BSD license above. Below is a list of certain publicly available software that is the source of intellectual property in YUI, along with the licensing terms that pertain to those sources of IP. This list is for informational purposes only and is not intended to represent an exhaustive list of third party contributions to the YUI.

- Douglas Crockford's JSON parsing and stringifying methods: In the JSON Utility, Douglas Crockford's JSON parsing and stringifying methods are adapted from work published at JSON.org. The adapted work is in the public domain.
- Robert Penner's animation-easing algorithms: In the Animation Utility, YUI makes use of Robert Penner's algorithms for easing.
- Geoff Stearns's SWFObject: In the Charts Control and the Uploader, YUI makes use of Geoff Stearns's SWFObject v1.5 for Flash Player detection and embedding. More information on SWFObject can be found here (<http://blog.deconcept.com/swfobject/>). SWFObject is (c) 2007 Geoff Stearns and is released under the MIT License (<http://www.opensource.org/licenses/mit-license.php>).

Index

Symbols

- `__after__()` method, 199
- `@authorize` decorator, 421–422
- `\` (back slash), in Mako, 68
- `__before__()` method, 198, 416–417
- `.build` directory, 272
- `##` characters, for comments, 67
- `${}` construct, for templates, 63, 67
- `<%def>` blocks (Mako), 76–77, 81
- `<%doc>` tag, 67
- `#document` element, 350
- `<form >` tag
 - action attribute, 91
 - method attribute, 93
- `/` (forward slash), template paths and, 64
- I18N abbreviation, 227
- `<%inherit>` tag, 92
- `__init__()` method, to set up classes, 150
- `__init__.py` file, 38
- `@jsonify` decorator, 361
- L10N abbreviation, 227
- `\n` (line-end) character (Unix), 27
- `<%namespace>` tag (Mako), 80–81
- `|` operator (Mako), 75–76
- `.pager()` template, variables to use as arguments
 - in, 188
- `\` (path separator) character (Windows), 27
- `/` (path separator) character (Linux and Mac OS X), 27
- `.po` (portable object) file, 229
- `.pot` (portable object template) file, 229
 - reload option (Paste HTTP server), 55
 - reload switch, starting server with, 166
- `__repr__` method, 150
- `@restrict` decorator, 181
- `\r\n` (line-end) character (Windows), 27
- `<%text>` tag, 67
- `-U` flag for commands, 21
- `__()` function, 228–230
- `_` (underscore) character, 66, 415
- `@validate` decorator, 103, 111–112
- 0.0.0.0 IP address, 36
- 127.0.0.1 IP address, 36
- 301 redirect to URL, 206
- 401 status code, 418
- 403 error document, 418, 437, 448–449
- 404 Not Found page
 - generation of, 445–446
 - SimpleSite and, 447–449

A

- abbreviations for internationalization and
 - localization, 227
- abort object, 47
- `abort()` function, 178, 412
- accessing objects
 - programmatically, 401–402
 - that aren't thread-safe, 294
- `AccountController` class, 439
- action attribute of `<form >` tag, 91
- actions, description of, 6, 38
- activate (activate.bat) script, 31, 490
- activating virtual Python environment, 18–19
- `add_fallback()` function, 239–240
- `__after__()` method, 199
- Ajax
 - description of, 325
 - JSON, 361–364
 - requests, debugging, 58, 360–361
 - SimpleSite and, 356–360
- Amazon S3, storing data in, 129–130
- `And` permission class, 426
- animation, adding to flash message, 354–356
- anonymous functions (JavaScript), 343
- Apache, proxying approach with
 - init scripts, creating, 496–497
 - log files, setting up, 496
 - overview of, 494–495
 - restarting stopped applications, 497
- `appconfig()` function, 402
- Apple Mac OS X, installation on, 24
- application component (WSGI)
 - as class instances, 371–372
 - overview of, 370–371
 - as Pylons controllers, 372–374
- application directory structure of project, 37–38
- application logs, server logs compared to,
 - 471–472
- application state, 404
- applications
 - See also* SimpleSite application
 - composite, 395–396
 - configuring, 51–52
 - constructing, Paste Deploy and, 394–395
 - creating
 - loading environment, 405
 - middleware chain, 406–407
 - with Paste Deploy, 402–403
 - PylonsApp instance, 406
 - developing on Windows, 27–28

- embedding into server
 - mod_wsgi tool, using, 497–502
 - overview of, 493
 - restarting stopped, 497
 - serving from installed environment, 492
 - WSGI, accessing programmatically, 401–402
- app_globals object, 50–51
- architecture
 - See also* Object-Relational API
 - egg entry points, websetup.py and, 391–392
 - history of, 390
 - SQLAlchemy
 - Declarative API, 160–162
 - Engine API, 136–138
 - Metadata and Type APIs, 138–141
 - overview of, 135–136
 - SQL Expression API, 141–146
- arguments used for sessions (SQLAlchemy), 152–153
- Array type (JavaScript), 338
- ASCII character set, 217
- assert keyword, 248
- assigning template variables using context c
 - global, 65–66
- Asynchronous JavaScript and XML. *See* Ajax
- attributes
 - of c object, 49
 - of request object, 44–45
- attrs argument, form helpers and, 97
- augmentation, 348
- authentication
 - See also* AuthKit
 - overview of, 415
 - security of
 - encrypting passwords, 431–432
 - SSL, 429–431
- authentication middleware (AuthKit)
 - configuring, 419–421
 - setting up, 418–419
- AuthKit
 - alternative authentication methods, 427–428
 - authentication middleware
 - configuring, 419–421
 - setting up, 418–419
 - authorization and, 421
 - authorization middleware, 422–423
 - @authorize decorator, 421–422
 - authorized() function, 423
 - components of, 417
 - controllers, protecting, 424
 - cookie-handling code, 427
 - description of, 417
 - drivers, 426
 - functional testing for controllers, 428
 - group functionality, 425
 - log messages and, 484
 - permission objects, @authorize decorator, 421–422
 - SimpleSite and
 - changing templates based on permissions, 438
 - controller actions, protecting, 436–438, 443–444
 - middleware, setting up, 433–434
 - signing in and out, 439–440
 - styling sign-in screen, 441–442
 - User Management API, 445
 - websetup.py, adjusting, 434–436
 - source code, as basis for customization, 418
 - SQLAlchemy driver, 434
 - authkit.cookie.signoutpath option, 421
 - authorization
 - See also* AuthKit
 - groups, roles, and, 424–426
 - overview of, 415
 - restricting access using _before_() method, 416–417
 - authorization middleware (AuthKit), 422–423
 - authorized() function (AuthKit), 423, 438
 - autodoc extensions (Sphinx), 276

B

 - Babel extractors, 236–237
 - Babel, using, 230–232
 - back slash (\), in Mako, 68
 - Bangert, Ben, 390
 - base controller for SimpleSite, role of, 176–177
 - base template for SimpleSite, creating, 169
 - base/index.html file, <body> part of, 332
 - BaseController class, 176, 372
 - Beaker package
 - caching functionality, 87
 - description of, 29
 - session handling and, 190
 - sessions, 152
 - Becker, Joe, 218
 - BEGIN keyword (SQLAlchemy), 154
 - best practice for Routes use, 209–211
 - bin directory, 31
 - binary numbers, 217
 - bit, 217
 - bleeding edge, working on, 22
 - body() def (Mako), 81
 - Boolean data type (JavaScript), 341
 - boto package, 130
 - breadcrumbs for SimpleSite, 319–321
 - browser CSS, resetting, 326–327
 - browser detection, feature detection compared to, 354
 - browsers
 - event handling in, 352–353
 - Internet Explorer 6, downloading files as attachments and, 101
 - Opera, 243

BSD, installation on, 23
 bubbling-up model of event handling, 353
 Buildout tool, 15, 488–489
 built-in types supported by SQLAlchemy, 139

C

C and C++ extensions, support for, 23
 c object, 49–50
 C:\Python25\Scripts directory, 25–27
 CacheMiddleware component (WSGI), 404, 410
 cache_dir variable, 78
 caching
 Beaker package and, 87
 Mako templating language and, 77–79
 callAjax() function, 359, 362
 capture() function (Mako), 71, 79–80
 capturing output (Mako), 79–80
 Cascade component (WSGI), 404, 408
 cascade object, 167
 Cascading Style Sheets (CSS)
 adding to form, 102
 of DOM elements, changing, 351
 fonts, 327–328
 grid framework
 nested grids, 331
 overview of, 328–329
 special nested grids, 331–332
 template preset grids, 329–331
 updating SimpleSite to use, 332–336
 reducing page load time, 365–366
 resetting browser, 326–327
 case sensitivity of URL, 205
 century routing variable, 201
 certificate-signing request for SSL, 429
 CGI script (Python)
 deploying application as, 493
 overview of, 4
 pros and cons of, 5
 Pylons techniques compared to, 5
 cgi.FieldStorage object, 99
 cgith module, 382
 chained validators, 121, 296
 changing URL, 206
 character set, ASCII, 217
 chr() function, 217
 class attribute, specifying, 97
 class instances, WSGI applications as, 371–372
 classes
 AccountController, 439
 And permission, 426
 BaseController, 176, 372
 FancyValidator, 115
 GzipMiddleware
 custom start_response callable, 384–385
 final version of code, 386–387
 start_response() callable, 385–386
 testing, 387–388
 HomegrownController, 416–417

Invalid exception, 114
 meta.Session, 177
 NavController, 303
 SQLAlchemy), 150–151
 StackedObjectProxy, 409
 yui-gf, 331
 clean separation, 7
 code
 See also listings; source code
 cookie-handling (AuthKit), 427
 to implement WSGI server API, 376–377
 logic, separating view code and, 72
 make_app() function, 403–404
 start_response() callable
 custom, to return GzipFile object, 384–385
 final version of, 386–387
 updating GzipMiddleware class to call, 385–386
 using run_with_cgi() to run hello() WSGI application, 377
 code pages, 218
 code points, 218
 codecs module, 223
 Collins, Lee, 218
 column widths, fixed, 329
 command line, testing from, 261–262
 command prompt, accessing in Windows, 26
 command-line debugging in nose, 250–251
 command-line options, virtualenv.py tool, 19
 commands
 for installation, 13–14
 nosetests, 254, 260
 paster create, 31–32
 paster make-app, 491
 paster make-config, 491
 paster serve, 393
 paster setup-app, 173, 392
 python, 23
 register, 460
 sphinx-build, 271
 -U flag, 21
 Windows and, 16
 Comment object (SQLAlchemy), 151
 comments, 67, 263–264
 comments system
 controller for
 creating, 281–282
 planning, 280
 updating to handle comments, 282–285
 overview of, 279–280
 routes for, modifying, 280–281
 setting page ID automatically, 285–288
 committing session (SQLAlchemy), 152
 community support, 9
 compiling Python directly from source, 23
 components, 9
 composite applications, 395–396
 concatenating strings, 145
 conditions argument (Routes), 212–215

- config directory, 37
- config file
 - application, constructing, 394–395
 - composite applications, 395–396
 - default options, 393
 - defining default language in, 237
 - factories
 - alternative ways to specify, 400
 - overview of, 397–399
 - inheritance and, 400
 - overview of, 392–393
 - pipelines and filters, 396–397
 - server, constructing, 393–394
- config object, 47, 51
- config/environment.py file, editing, 64
- config/middleware.py file, `make_app()`
 - function, 403–404
- config/routing.py file
 - changing, 314–315
 - `navigation_from_path()` function, 316–317
 - overview of, 196
- configuration files, `development.ini`, 33–34
- configuration options, accessing
 - programmatically, 402
- configuring
 - applications, 51–52
 - authentication middleware, 419–421
 - engine for SimpleSite, 171
 - logging
 - formatter sections, 476–477
 - handler sections, 476
 - logger sections, 475–476
 - `setup.py` file for SimpleSite
 - dependencies, 453–454
 - extra dependencies, 454–455
 - extra dependency links, 455
 - `long_description` argument, 456–457
 - metadata, specifying, 455–456
 - overview of, 452
 - production config file template, 457–459
 - version number, choosing, 453
 - validators, 107–108
- `ConfirmType` validator, 107
- `connect()` function (routing map), 201
- Connection object (SQLAlchemy), 136
- connection pools, 137
- console object (Firebug), 362
- constraining tag names, 293–294
- content
 - context c, assigning template variables using, 65–66
 - describing with URL, 205
 - streaming, 412–413
- context object
 - Mako, 71–72
 - `tmpl_context`, 49–50
- control structures in Mako, 67
- controller actions
 - calling with routing variables, 198–199
 - protecting for SimpleSite, 436–438, 443–444
- controllers
 - See also* page controller
 - actions of, making private, 415–416
 - for comments system
 - creating, 281–282
 - planning, 280
 - updating to handle comments, 282–285
 - creating, 38–39
 - decorators available for, 52
 - description of, 6
 - for forms, 92
 - functional testing for AuthKit, 428
 - for navigation hierarchy, creating, 303–311
 - pagetag, 295–298
 - protecting, 424
 - SimpleSite example
 - base, role of, 176–177
 - `create()` method, 180–183
 - creating, 167–168
 - customizing, 167
 - `delete()` method, 185–186
 - edit and save() methods, 183–184
 - `list()` method, 185
 - `new()` method, 178–180
 - updating to support editing pages, 177
 - `view()` method, 178
 - tag, creating, 291–293
 - types of, 52
 - WSGI applications as, 372–374
 - `WSGIController`, 411
- controllers directory, 37
- `controllers/errors.py`, 445–446
- `controller_scan()` function
 - description of, 198
 - returning list of valid controllers using, 200
- controlling
 - propagation using loggers
 - filtering messages, 482
 - options, 483
 - overview of, 480–482
 - which messages are logged using handlers, 480
- convention over configuration, 6
- cookie-handling code (AuthKit), 427
- copying temp file data to permanent location, 99
- `copytree()` function (shutil module), 129
- `create()` method, updating controller to support
 - editing pages, 180–183
- `create_engine()` function, 137
- Crockford, Douglas, 339
- cross-site scripting (XSS) attacks, 73
- CSS (Cascading Style Sheets)
 - adding to form, 102
 - of DOM elements, changing, 351
 - fonts, 327–328

- grid framework
 - nested grids, 331
 - overview of, 328–329
 - special nested grids, 331–332
 - template preset grids, 329–331
 - updating SimpleSite to use, 332–336
 - reducing page load time, 365–366
 - resetting browser, 326–327
 - customizing error documents for SimpleSite, 447–449
 - Cutrell, Edward, 205
- D**
- daemontools, 497
 - data
 - private, 415–416
 - querying, SimpleSite, 175–176
 - storing
 - in Amazon S3, 129–130
 - approaches to, 127
 - in databases, 130–132
 - in filesystems, 127–129
 - data directory, 37, 127
 - data persistence layer, RDBMS and, 132
 - data source name, specifying for database, 171
 - database tables for SimpleSite, creating, 173–175
 - databases
 - See also* RDBMS
 - creating with SQLite, 135
 - object-relational principles, 146–148
 - storing data in
 - object databases, 131
 - overview of, 130
 - XML databases, 131–132
 - Unicode and, 225
 - DateConverter validator, 106–108
 - dates in SimpleSite, formatting, 189–190
 - datetime.datetime objects, 189–190
 - Davis, Mark, 218
 - DB-API drivers
 - installing, 134–135
 - for popular RDBMS systems, 133–134
 - deactivating virtual Python environment, 18–19
 - debug messages in nose, 249
 - debug mode in INI file, turning off, 412
 - debugging
 - Ajax requests, 360–361
 - interactive debugger
 - description of, 55
 - enabling error reporting, 59
 - in production environments, 59
 - tabs, 56–58
 - Declarative API (SQLAlchemy), 160–162
 - decoding
 - request parameters, 224
 - Unicode, 221
 - decorator tool, 29
 - decorators, 52, 361
 - def (Mako), 71
 - def blocks (Mako), 76–77, 81
 - default language, defining in config file, 237
 - default variables (Routes), 201–202
 - DELETE statement (SQLAlchemy), 145
 - delete() action, protecting, 437
 - delete() method, updating controller to support
 - editing pages, 185–186
 - deleted pages, handling, 289–290
 - deleting
 - pages, 298
 - tags, 298
 - dependencies
 - overview of, 29–31
 - for SimpleSite, configuring, 453–455
 - deployment
 - Apache proxying
 - init scripts, creating, 496–497
 - log files, setting up, 496
 - overview of, 494–495
 - restarting stopped applications, 497
 - embedding application into server, 493, 497–502
 - options for, 492–493
 - overview of, 487
 - proxying, overview of, 494
 - steps in, 487
 - system Python environment and, 488
 - virtual Python environment and
 - activate script, 490
 - application instance, setting up, 491
 - Buildout, 488–489
 - config file, creating, 491
 - installing software to, 490–491
 - serving application, 492
 - setting up, 489
 - on Windows, 502
 - derived/nav/create_page.html template, 318
 - derived/nav/create_section.html template, 318
 - detailed errors in nose, 249
 - developing applications on Windows, 27–28
 - development mode (setuptools package), 174
 - development process, types of testing and, 246–247
 - development.ini file
 - cache_dir variable, 78
 - description of and code for, 33
 - host option in, 36
 - options, 34
 - Dialect object (SQLAlchemy), 137
 - dictionaries
 - environ, 379–381
 - JavaScript, 345
 - request.environ, 43
 - request.urlvars, 199, 440
 - directory, 128. *See also* specific directories
 - directory structure of project, 36–37

- disabling
 - implicit defaults feature, 209
 - interactive debugger, 59
 - route memory, 208
 - route minimization feature, 207
- disambiguated URL, 206
- dispatching, 198–199
- distribution, packaging project for (SimpleSite)
 - egg file, building, 459–460
 - egg file, publishing on PyPI, 460–462
 - overview of, 459
- distutils.cfg file, virtual Python environments
 - and, 15
- Dive Into Python (Pilgrim), 8
- Django
 - clean separation, 7
 - Pylons compared to, 6, 390
- <%doc> tag, 67
- docs directory, 37, 271–272
- docstrings
 - help() function and, 265
 - tools working on, 265
 - use of, 264
- doctest library, 246
- doctests, 266–268
- #document element, 350
- Document Object Model (DOM)
 - manipulating, 352
 - navigating, 351–352
 - overview of, 349
 - parse tree, 350
- documentation
 - See also* documentation tools
 - doctests, 266–268
 - overview of, 263
 - reStructuredText language, 268–270
 - Sphinx and
 - automatically generating documentation
 - with, 276–277
 - Python source code, documenting with,
 - 273–274, 276
 - SimpleSite project, documenting with,
 - 270–273
 - syntax highlighting, 277–278
- documentation tools
 - comments, 263–264
 - docstrings, 264
 - help() function, 265–266
- docutils package, 268–270
- docutils.core.publish_pars() function, 270
- DOM. *See* Document Object Model
- downloading from Python Package Index, 14
- drivers
 - AuthKit, 426
 - DB-API, 133–135
- drop-down lists, populating one from values of
 - another, 356–360
- Durus object database, 131
- dynamic parts of routes, 195, 200

E

- e-mail address, validating, 101–102
- E-Tag caching for static files, 36
- easy_install program
 - choosing package versions with, 19–20
 - description of, 31
 - troubleshooting, 21–22
 - working with, 16
- easy_install.pth file, 20
- Eby, Philip J., 369
- echo option (SQLAlchemy), 143
- ECMAScript, 340. *See also* JavaScript
- edit() action
 - protecting, 436
 - updating to handle comments, 283
- edit() method, updating controller to support
 - editing pages, 183–184
- editing
 - config/environment.py file, 64
 - pages, updating controller to support
 - create() method, 180–183
 - delete() method, 185–186
 - edit and save() methods, 183–184
 - list() method, 185
 - new() method, 178–180
 - overview of, 177
 - view() method, 178
- editors, 28
- egg entry points, 391–392
- egg file
 - building, 459–460
 - publishing on PyPI, 460–462
- eggs
 - description of, 18
 - installing directly with Easy Install, 20
- Email validator, 104–106
- EmailForm schema, 104
- embedding application into server
 - mod_wsgi tool, using, 497–502
 - overview of, 493
- enabling
 - error reporting in interactive debugger, 59
 - Firebug plug-in, 336
- encode() method (Unicode), 222
- encoding
 - code points into binary numbers, 218
 - Unicode, 222
- encrypting passwords, 431–432
- Engine API (SQLAlchemy), 136–138
- engine for SimpleSite, configuring, 171
- engine.connect() function, 137
- engine_from_config() function, 170
- entities, 146
- environ argument, 199, 202
- environ dictionary
 - description of, 379
 - modifying, 380–381

- environment, loading when creating
 - application, 405
- environment variables
 - HTTP headers, 42
 - set for all requests, 42
 - viewing, 43–44
 - WSGI, 43
- environment.py file, 51
- equality operators (JavaScript), 340
- error documents, 412
- error handling, Unicode, 220–221
- error messages
 - formatting of, and HTML Fill tool, 110
 - FormEncode tool, 105
 - 403 Forbidden, 418, 437, 448–449
 - 404 Not Found page
 - generation of, 445–446
 - SimpleSite and, 447–449
 - highlighting in red, 102
 - including full path and, 32
 - Missing value, customizing, 120
 - pkg_resources.ExtractionError, 21
 - styling to appear in red, 182
 - Unicode, 217
- error reporting, enabling in interactive
 - debugger, 59
- ErrorHandler component (WSGI), 404, 410
- ErrorHandler middleware, 55
- errors in WSGI application, handling, 381–382
- escape functions (Mako), 75
- escape() function, 73–74
- Event Model
 - browser detection vs. feature detection, 354
 - overview of, 352–353
 - same origin policy, 354
- exception handling, 412
- eXist XML database, 131–132
- explicit routes, 202, 210
- expressing file size in human-readable terms, 129
- Extra Data tab (interactive debugger), 56
- extracting
 - internationalizable messages with Babel
 - extractors, 236–237
 - messages, 229
- extra_envron argument, 428
- ez_setup.py file, 37

F

- factories
 - alternative ways to specify, 400
 - overview of, 397–399
- fallback languages, 239–240
- FancyValidator class, 115
- feature detection, browser detection compared to, 354
- field helper, 179
- fields.html file for nav table, 305

- file extensions in URL, 206
- FileHandler, 478
- files
 - See also specific files*
 - logging to, 478
 - naming, 100
 - uploading, 98–101
 - writing Unicode data to, 223
- filesystem, storing data in, 127–129
- file_field helper, 98
- filter functions (Routes), 215–216
- filter() and filter_by() methods, 157–158
- filtering messages using propagation, 482
- filters
 - applying in view templates, 75–76
 - overview of, 396–397
 - WSGI, accessing programmatically, 401–402
- Firebug
 - console object, 362
 - description of, 336
 - Inspect button, 351
 - Net tab, 360
 - testing JavaScript in, 337
- Firefox
 - attachment download dialog box, 101
 - web browser, LiveHTTPHeaders extension, 39
- fixed column widths, 329
- flash message system, adding animation to, 354–356
- flash messages, using, 190–192
- flushing session (SQLAlchemy), 152, 155
- fonts, specifying, 327–328
- footer def
 - adding link enabling users to add new tags, 298
 - for nav controllers, 310
 - templates/derived/page/view.html template, 289
- footers, updating, 186, 443–444
- Forbidden status code, 418, 437, 448–449
- ForEach validator, 116, 120
- foreign key
 - description of, 147
 - one-to-many mappings and, 279
- form and cookie authentication method, 419
- form() helper, 96
- <form > tag
 - action attribute, 91
 - method attribute, 93
- formatter sections, 476–477
- formatters, 110
- formatting dates and times, 189–190
- FormBuild package, 179
- FormEncode package, 29, 184
- FormEncode schema
 - creating, 180
 - for nav controller, 303

FormEncode tool

See also HTML Fill tool

chained validators, 121
 EmailForm schema and, 105
 error messages produced by, 105
 nestedvariables module, 118
 parts of, 103
 prevalidators, 119
 validation schema and, 104
 validators

 configuring, 107–108
 custom, creating, 112–115
 list of, 106–107
 options supported by, 107

forms

 building with helpers, 96–98

 controllers for, 92

 handling

 approaches to, 91
 manually, 101–103

 overview of, 91

 repeating fields problem

 complete code for, 122–124
 controller code for, 121
 field names for, 119
 overview of, 115
 role field values, 116
 schema for, 116–121
 solutions to, 116
 testing, 124–125

 request.params object and, 93

 resubmitted data problem, 95–96

 simple template for, 92

 submitting using GET or POST HTTP

 methods, 93, 95
 uploading files, 98–101
 validation schema for, 103

forward slash (/), template paths and, 64

403 Forbidden message, 418, 437, 448–449

404 Not Found page message

 generation of, 445–446
 SimpleSite and, 447–449

FTP (File Transfer Protocol), 39

function option (conditions argument), 212

function scope and closures (JavaScript),
 343–344

functional testing

 for controllers, 428

 description of, 246

 of objects, 260–261

 page controller save() action, 257–260

 with paste.fixture

 documentation on, 261
 nosetests command, 254
 overview of, 252–253
 test.ini file, 254–256

functions

See also helper functions

abort(), 178, 412
 add_fallback(), 239–240
 anonymous (JavaScript), 343
 appconfig(), 402
 authorized(), 423, 438
 callAjax(), 359, 362
 capture(), 71, 79–80
 chr(), 217
 connect(), 201
 controller_scan(), 198, 200
 copytree(), 129
 create_engine(), 137
 docutils.core.publish_pars(), 270
 engine.connect(), 137
 engine_from_config(), 170
 escape(), 73–75
 generating for routes, 202–203
 getattr(), 50
 get_lang(), 233
 getLogger(), 474
 hasattr(), 50
 hasOwnProperty(), 348
 help(), 265–266
 h.size_to_human(), 129
 h.url_for()
 description of, 48
 generating routes with, 202–203
 generating URLs with, 195
 referencing static resources with, 204
 init_model(), 173
 JavaScript, 338, 343
 lazy_ugettext(), 241
 link(), 77
 loadapp(), 401–402
 load_environment(), 405
 loadfilter(), 401
 loadserver(), 401
 make_app(), 395, 402–407
 make_app(), 196
 message, 107
 navigation_from_path(), 315–317
 navigation_links(), 77
 now(), 149
 object(), 347
 ord(), 217, 220
 os.listdir(), 223
 os.stat(), 128
 redirect_to, 47, 202–203, 412
 relation(), 151
 render(), 65, 69–70, 85–86, 110–111
 render_body(), 78
 render_mako(), 86
 render_signin(), 442
 render_template(), 86
 scoped_session(), 153

- `server_runner()`, 394
- `sessionmaker()`, 152
- `set_lang()`, 239
- `setup_app()`, 174, 255–256
- `test_save_prohibit_get()`, 258
- `time.sleep()`, 413
- `time_ago_in_words()`, 129
- `ungettext()`, 241
- `unichr()`, 220
- `webhelpers.html.escape`, 73
- in `YAHOO.lang`, 341–342

G

- generating routes, functions for, 202–203
- Genshi templating language, 88–89
- GET method, 40, 93–95
- `get(key, default)` on `request.params` object, 93
- `getall()` method, 93
- `getattr()` function, 50
- `getLogger()` function, 474
- `getone()` method, 93
- `get_lang()` function, 233
- globals
 - See also* request object; response object
 - abort, 47
 - `app_globals` object, 50–51
 - c context, assigning template variables using, 65–66
 - c object, 49–50
 - config, 47, 51
 - description of, 46, 409
 - h object, 48–49
 - `redirect_to`, 47
 - session, 48
- GNU `gettext`, 229
- `go-pylons.py` script, 22
- `greeting.html` template, 64
- grids
 - nested, 331
 - special nested, 331–332
 - style sheet, 328–329
 - template preset, 329–331
 - updating `SimpleSite` to use, 332–336
- groups, authorization and, 424–426
- Guan, Zhiwei, 205
- Gzip compression, 383–384
- `GzipMiddleware` class
 - custom `start_response()` callable, 384–385
 - final version of code, 386–387
 - `start_response()` callable, 385–386
 - testing, 387–388

H

- handler sections, 476
- handlers
 - capturing `AuthKit` messages using, 484
 - controlling which messages are logged using, 480

- redirecting log output using
 - logging to files, 478
 - logging to `wsgi.errors`, 479–480
 - overview of, 477
- handling errors, Unicode, 220–221
- handling requests
 - Cascade, 408
 - middleware chain, 410
 - overview of, 407–408
 - `PylonsApp` instance, 410–411
 - `RegistryManager`, 409
 - `WSGIController`, 411
- handling responses
 - error documents, 412
 - exception handling, 412
 - overview of, 411–412
 - returning Unicode from action, 413
 - streaming content, 412–413
- handling translations for internationalization, 229
- hard-coded variables (`Routes`), 201
- `hasattr()` function, 50
- `HasAuthKitGroup` permission, 425
- `HasAuthKitRole` permission, 425
- hash tables, 345
- `hasOwnProperty()` function (JavaScript), 348
- headers in piece of middleware, changing, 381
- helloworld application directory structure, 37–38
- `HelloWorld` directory structure, 36–37
- `help()` function, 265–266
- helper functions
 - built-in, building forms with, 96–98
 - description of, 48–49
 - field, 179
 - `stylesheet_link()`, 182
 - writing to use HTML literals, 74–75
- hierarchy of named loggers, 473
- history
 - of `Pylons`, 389–390
 - of Unicode, 217–218
- home page when signed out, 440
- `HomegrownController` class, 416–417
- hosts, running project with web server, 36
- `h.size_to_human()` function, 129
- HTML (Hypertext Markup Language)
 - JavaScript in, 349
 - overview of, 39
 - templating system and, 63
- HTML escaping, `Mako` and, 73
- HTML Fill tool
 - controller for, 109
 - customizing call for, 112
 - description of, 103, 108
 - error message formatting, 110
 - generated example, 109
 - `render()` function, 110–111
- HTML view, `Inspect` in DOM tab, 351

- HTTP (Hypertext Transfer Protocol)
 - overview of, 39
 - requests, 40–41
 - specification, 41
 - status codes, 41
- HTTP digest authentication, 427, 429
- HTTP GET request, 40
- HTTP headers
 - environment variables and, 42
 - Pylons compared to WSGI, 370
- HTTP response with X-Debug-URL header, 58
- HTTP response status, Pylons compared to WSGI, 370
- HTTP status codes, authentication, authorization, and, 418
- HTTPException, 412
- HTTPS, port 443 and, 431
- HTTP_ACCEPT_LANGUAGE header, 240
- HTTP_PROXY environment variable, 20
- h.url_for() function
 - description of, 48
 - generating routes with, 202–203
 - generating URLs with, 195
 - referencing static resources with, 204
- Hypertext Markup Language (HTML)
 - JavaScript in, 349
 - overview of, 39
 - templating system and, 63
- Hypertext Transfer Protocol (HTTP)
 - overview of, 39
 - requests, 40–41
 - specification, 41
 - status codes, 41
- hyphens in URL, 205
-
- I18N abbreviation, 227
- id column, adding to each table as primary key, 147
- id field, 280
- IDLE editor, 28
- implicit defaults feature, 209
- importing
 - functions into templates, 80–81
 - helper function, 48
 - HTML Fill tool, 109
 - validate() decorator, 111
- index.html file, 36
- index.txt file, 271
- <%inherit> tag, 92
- inheritance
 - configuring validators using, 108
 - in JavaScript, 339, 347–348
 - Paste Deploy and, 400
 - in SQLAlchemy, 299–301
- inheritance chains (Mako)
 - next namespace, 83–85
 - overview of, 81
 - parent namespace, 85
 - simple inheritance, 82–83
- init scripts, creating for Apache proxying, 496–497
- init_model() function, 173
- injection attacks, 142–143
- INSERT statement (SQLAlchemy), 154
- inspecting DOM element, 351
- inspect_call() method, 412
- installation
 - bleeding edge, working on, 22
 - easy_install program
 - choosing package versions with, 19–20
 - troubleshooting, 21–22
 - working with, 16
 - eggs, 18
 - on Linux and BSD, 23
 - on Mac OS X, 24
 - overview of, 13
 - with proxy server, 20
 - of Pylons, 17
 - Python Package Index, 14
 - quick, on Linux, 13–14
 - virtual Python environment
 - activating and deactivating, 18–19
 - setting up, 14–15
 - virtualenv.py tool, setting options for, 19
 - on Windows, 24–26
- installing
 - Babel, 230
 - DB-API driver, 134–135
 - FormBuild package, 179
 - pysqlite2 module, 135
 - SQLAlchemy, 135
 - templating language, 88
 - YUI into public directory, 325
- interactive debugger
 - description of, 55
 - enabling error reporting, 59
 - in production environments, 59
 - tabs, 56–58
- interactive shell, testing and, 261–262
- internal static routes, 204
- internationalization
 - Babel extractors, 236–237
 - default language, defining in config file, 237
 - extracting messages and handling translations, 229
 - fallback languages, 239–240
 - lazy translations, 240–241
 - marking strings for, 228–229
 - overview of, 227
 - plural forms, 241–242
 - process of, 228
 - search engines and, 243
 - storing user language in sessions, 238–239

- TranslateDemo example
 - Babel, using, 230–232
 - overview of, 229–230
 - supporting multiple languages, 232–234
 - translations within templates, 235–236
 - updating message catalogs, 234–235
- Internet Explorer 6, downloading files as attachments and, 101
- introduction page, 167
- Invalid exception class, 114
- IP addresses, running project with web server, 36
- ISO 8859-1 encoding, 218

J

- JavaScript
 - See also* Ajax
 - Document Object Model and, 349
 - Event Model and, 352
 - function scope and closures, 343–344
 - functions, 343
 - in HTML, 349
 - inheritance in, 347–348
 - namespaces, 346–347
 - objects, 344–345
 - operators, 340
 - overview of, 338–339
 - prototypes, 348–349
 - reducing page load time, 365–366
 - testing in Firebug, 337
 - this, 345–346
 - types, 341–342
- JavaScript: The Good Parts* (Crockford), 339
- JavaScript web frameworks
 - description of, 325
 - most popular, 325
- YUI
 - adding to project, 325–326
 - animation classes, 354
 - event handling in, 353
 - fonts style sheet, 327–328
 - grids style sheet, 328–329
 - nested grids, 331
 - resetting browser CSS with, 326–327
 - special nested grids, 331–332
 - template preset grids, 329–331
 - updating SimpleSite to use grids, 332–336
- Jinja 1 templating language, 88–90
- JSON, 361–364
- @jsonify decorator, 361

K

- keys
 - foreign, 147, 279
 - primary, 146
 - private, for SSL, 429
- keywords
 - assert, 248
 - BEGIN (SQLAlchemy), 154

L

- L10N abbreviation, 227
- languages
 - See also* Mako templating language; Python language; templating languages
 - default, defining in config file, 237
 - fallback, 239–240
 - multiple, supporting, 232–234
 - reStructuredText, 268–270
- Latin-1 encoding, 218
- lazy_uggettext() function, 241
- length of URL, 205
- lib directory, 38
- lib/helpers.py module, 48
- lib/python2.5/site-packages directory, 29
- line-end characters (\n or \r\n), 27
- link() function, 77
- linkage pattern, 348
- Linux
 - installation on, 23
 - quick installation on, 13–14
- list comprehension, 158
- list() method, updating controller to support editing pages, 185
- list.html template
 - tag controller and, 292–293
 - updating to handle comments, 283–284
- listamatic web site, 322
- listings
 - accessing page object from attributes, 160
 - base controller for SimpleSite, 176
 - base template for SimpleSite, 169
 - CGI script example, 3
 - controllers/errors.py, 445–446
 - controllers/page.py file, 167
 - def block (Mako), 76
 - derived/form.html template, 117
 - derived/page/view.html template, updating footer, 186
 - development.ini file, 33
 - engine_test.py file, creating, 136
 - flushing session, 155
 - footer, updating to protect actions, 443–444
 - form to handle repeating fields problem, 122, 124
 - HomegrownController class, 416–417
 - inserting rows into tables, 159
 - list.html template, updating to use paginator, 187–188
 - long_description argument, customizing, 456
 - metadata_test.py file, creating, 138
 - model.py file
 - creating, 148–149
 - rewriting using Declarative API, 160–162
 - model_init.py, 171–173
 - object_test.py, creating, 152
 - page controller save() action, 257
 - production config file template, 457–459

- render_body() function, 78
 - route map, 196
 - setup_app() function, 255–256
 - SimpleSiteTemplate
 - creating, 462–463
 - using, 464–465
 - sqlexpression_test.py file, creating, 141
 - templates/derived/page/fields.html file, 178
 - test.ini file, 254
 - test_save() method, 259–260
 - YUI Rich Text Editor, setting up, 450
 - literal() objects, 73–74
 - literals, Unicode, 219–220
 - LiveHTTPHeaders extension (Firefox web browser), 39
 - LiveHTTPHeaders, GET and POST requests in, 93
 - loadapp() function, 401–402
 - load_environment() function, 405
 - loadfilter() function, 401
 - loading environment when creating applications, 405
 - loadserver() function, 401
 - localization
 - See also* internationalization
 - fallback languages, 239–240
 - lazy translations, 240–241
 - overview of, 227
 - plural forms, 241–242
 - process of, steps in, 228
 - search engines and, 243
 - log files, setting up for Apache proxying, 496
 - log messages
 - AuthKit and, 484
 - capturing SQLAlchemy messages using propagation, 483–484
 - handling
 - controlling which messages are logged, 480
 - logging to files, 478
 - logging to wsgi.errors, 479–480
 - overview of, 477
 - log statements, writing into code, 471
 - logger sections, 475–476
 - loggers, controlling propagation with
 - filtering messages, 482
 - options, 483
 - overview of, 480–482
 - logging
 - production configuration file and, 485
 - types of logs, 471–472
 - logging configuration
 - formatter sections, 476–477
 - handler sections, 476
 - logger sections, 475–476
 - logging module
 - log levels, 473–474
 - overview of, 472–473
 - templates and, 474–475
 - variables, 474
 - logic code, separating view code and, 72
 - long_description argument, customizing, 456–457
 - loose coupling, 7
 - lowercase URL, 205
- ## M
- Mac OS X, installation on, 24
 - main links for SimpleSite, 319–321
 - maintaining performance, SQLAlchemy and, 162–163
 - make_app() function
 - code, 403–404
 - description of, 402
 - loading environment, 405
 - middleware chain, 406–407
 - PylonsApp instance, 406
 - syntax, 395
 - make_map() function, 196
 - Mako templating language
 - applying filters in templates, 75–76
 - basic syntax, 66–69
 - body() def, 81
 - built-in escape functions, 75
 - caching, 77–79
 - capturing output, 79–80
 - def blocks, using, 76–77, 81
 - description of, 6–7, 30
 - example of, 63
 - inheritance chain features, 168
 - inheritance chains
 - next namespace, 83–85
 - overview of, 81
 - parent namespace, 85
 - simple inheritance, 82–83
 - <%namespace> tag, 80–81
 - runtime built-ins, 70–72
 - security considerations, 73–74
 - separating logic code and view code, 72
 - Unicode and, 225
 - working with Jinja, 89–90
 - writing helpers to use HTML literals, 74–75
 - mako-render script, 31
 - MANIFEST.in file, 37
 - manipulating Document Object Model, 352
 - many-to-many mappings, 290. *See also* tags
 - many-to-many relationship, 147
 - map.connect(), naming routes and, 195
 - Mapper object (Routes), 197
 - mappers (SQLAlchemy), 150–151
 - mapping
 - root URL to controller action, 39
 - URL, 195
 - marking strings for internationalization, 228–229
 - matching URL, 197
 - Mercurial repository, 22
 - message catalogs, updating, 234–235
 - message function (validators), 107

- messages, in internationalization terminology, 228
 - meta.Session class, 177
 - Metadata API (SQLAlchemy), 138–141
 - metadata for SimpleSite, specifying, 455–456
 - MetaData object, 173
 - metadata of database, describing (SQLAlchemy), 148–150
 - method attribute of <form> tag, 93
 - method option (conditions argument), 212
 - methods
 - `__after__()`, 199
 - `__before__()`, 198, 416–417
 - of context object (Mako), 71–72
 - `create()`, 180–183
 - `delete()`, 185–186
 - `edit()`, 183–184
 - `encode()` (Unix), 222
 - `filter()` and `filter_by()`, 157–158
 - form and cookie authentication, 419
 - `GET`, 40, 93–95
 - `getall()`, 93
 - `getone()`, 93
 - `init()`, 150
 - `inspect_call()`, 412
 - `list()`, 185
 - log levels and, 473
 - for manipulating DOM, 352
 - `nav_to_path()`, 319
 - `new()`, 178–180
 - `POST`, 40, 93–95
 - `_repr_`, 150
 - of request object, 44–45
 - `request.params.getall()`, 45
 - `request.params.getone()`, 45
 - `save()`, 183–184
 - `session.commit()`, 156
 - `session.execute()`, 156
 - `strftime()`, 189–190
 - `test_save()`, 259–260
 - `test_save_invalid_form_data()`, 259
 - `test_save_invalid_id()`, 258
 - `to_python`, 105
 - `validate_python()`, 113
 - `view()`, 178, 295
 - Microsoft
 - See also* Windows
 - Internet Explorer 6, downloading files as attachments and, 101
 - middleware
 - ErrorHandler, 55
 - setting up for SimpleSite, 433–434
 - middleware component (WSGI)
 - advantages of, 389
 - building stack out of, 389–390
 - overview of, 369, 378–379
 - testing, 387–388
 - writing
 - environment, modifying, 380–381
 - errors, handling, 381–382
 - overview of, 379–380
 - response, altering, 383–387
 - status and headers, changing, 381
 - middleware stack, 403–404
 - middleware.py file, 51
 - Missing value message, customizing, 120
 - model directory, 38, 128
 - model layer, storing data and, 127
 - Model View Controller architecture, 5–6
 - models
 - for SimpleSite
 - creating, 171–173
 - engine, configuring, 171
 - overview of, 170
 - separating from templates, 176
 - modifying Routes system, 38–39
 - module-level blocks, 69
 - mod_wsgi tool
 - description of, 493
 - embedding with, 497–499
 - troubleshooting, 501–502
 - virtual host, setting up, 499–500
 - Mozilla Firefox, attachment download dialog box, 101
 - MultiDict object, 45
 - multiple languages
 - supporting, 232–234
 - working with, 89–90
 - multiprocess approach to deployment, 492–493
 - multithreading approach to deployment, 492
 - MVC (Model View Controller) architecture, 5–6
 - MySQL, pool_recycle option, 171
 - MySQLdb module, 134
- ## N
- `\n` (line-end) character (Unix), 27
 - name argument, form helpers and, 97
 - named routes
 - description of, 203–204
 - filter functions and, 215–216
 - <%namespace> tag (Mako), 80–81
 - namespaces
 - JavaScript, 346–347
 - Mako, 71, 83–85
 - YAHOO, 346
 - naming
 - files, 100
 - routes, 195
 - routing variables, 202
 - table columns, 151
 - Nav object, mapper for, 300
 - nav table, 299
 - NavController class, 303
 - navigating Document Object Model, 351–352

- navigation hierarchy for SimpleSite
 - controllers, creating, 303–311
 - creating, 299
 - elements, adding, 319–321
 - inheritance in SQLAlchemy and, 299–301
 - initial data, setting up, 301–303
 - page controller, 311–313
- navigation_from_path() function, 315–317
- navigation_links() function, 77
- navigator tool, 187
- nav_to_path() static method, 319
- nested grids, 331–332
- nestedvariables module (FormEncode tool), 118
- new() method, updating controller to support editing pages, 178–180
- NewCommentForm schema, 282
- NewNavForm schema, 304–305
- NewPageForm schema, 311
- NewTagForm schema
 - UniqueTag validator, 293
 - updating, 291
- next namespace (Mako), 83–85
- nodes, 353
- nopage() action, 317
- nose tool
 - command-line debugging, 250–251
 - debug messages, 249
 - detailed errors, 249
 - overview of, 30, 247–249
 - paste.fixture and, 252
 - search locations, 251
- nosection() action, 317
- nosetests command, 254, 260
- nosetests script, 31
- now() function (SQLAlchemy), 149
- Null data type (JavaScript), 341
- Number data type (JavaScript), 341
- numbers, in JavaScript, 340

O

- object databases, 131
- object() function, 347
- Object-Relational API (SQLAlchemy)
 - classes and mappers, 150–151
 - database metadata, describing, 148–150
 - objects, 159–160
 - overview of, 146
 - queries, 157–159
 - sessions
 - examples of, 153–156
 - overview of, 152–153
- object-relational mappers (ORMs)
 - See also* SQLAlchemy
 - advantages and disadvantages of, 132
 - available for Python, 133
- object-relational principles, 146–148

objects

- See also specific objects*
- accessing
 - programmatically, 401–402
 - that aren't thread-safe, 294
- docstrings, 264
- functional testing of, 260–261
- JavaScript, 344–345
- SQLAlchemy, 159–160
- OneOf validator, 117
- one-to-many mappings, 279
- one-to-many relationship, 147
- one-to-one relationship, 147
- 127.0.0.1 IP address, 36
- Online Assistance box (interactive debugger), 57
- Open Command Here Powertoy, 26
- Opera web browser, 243
- operators
 - JavaScript, 340
 - Python, 144
- Oracle Berkeley DB XML database, 131–132
- ord() function, 217, 220
- ORDER_BY clause (SQLAlchemy), 145
- origins, 354
- ORMs (object-relational mappers)
 - See also* SQLAlchemy
 - advantages and disadvantages of, 132
 - available for Python, 133
- os.listdir() function, 223
- os.path module, 128
- os.stat() function, 128
- output buffering (Mako), 79–80
- output encoding (Unicode), 225

P

- packages
 - See also* Beaker package; Paste Deploy package; setuptools package; WebHelpers package
 - boto, 130
 - docutils, 268–270
 - FormBuild, 179
 - FormEncode, 29, 184
 - Paste, 130
 - pkg_resources, 392
 - Pygments, 277
 - Pylons-0.9.7-py2.5.egg, 30
 - removing, 20
 - simplejson, 30
 - upgrading, 20
 - versions, choosing with Easy Install, 19–20
 - WebError, 30
 - WebOb, 31, 44, 46
 - with extensions, 23
- packaging SimpleSite for distribution
 - egg file, building, 459–460
 - egg file, publishing on PyPI, 460–462
 - overview of, 459

- page controller
 - for navigation hierarchy, 311–313
 - for SimpleSite, creating, 167–168
 - save() action
 - listing, 257
 - testing, 257–260
- page ID, setting automatically, 285–288
- page load time, reducing, 365–366
- Page view for SimpleSite
 - deleted pages, handling, 289–290
 - updating, 288–289
- page widths, setting, 329
- pageargs dictionary (Mako), 71
- pageid field, 280
- pages, deleting, 298
- pagetag controller, 295–298
- paginate module (WebHelpers), 187
- pagination, using, 186–189
- parent namespace (Mako), 85
- passing request-specific information to parts of code using context object, 49–50
- passwords, encrypting, 431–432
- Paste Deploy package
 - accessing objects programmatically, 401–402
 - application, constructing, 394–395
 - config file and, 392
 - creating applications with, 402–403
 - factories
 - alternative ways to specify, 400
 - overview of, 397–399
 - inheritance and, 400
 - pipelines and filters, 396–397
 - server, constructing, 393–394
- Paste HTTP server
 - IP addresses and, 36
 - reload option, 55
 - running project with, 34–35
 - SSL and, 430
- Paste package, 30
- Paste Script developer documentation, 393
- paste.fixture
 - documentation on, 261
 - nosetests command, 254
 - overview of, 252–253
 - response object, 253
 - test.ini file, 254–256
- paste.testing_variables dictionary, 260
- paster create command, 31–32
- paster make-app command, 491
- paster make-config command, 491
- Paster project template, making SimpleSite into
 - overview of, 462–465
 - variables, 465–468
- paster script, 31
- paster serve command, 393
- paster setup-app command, 173, 392
- PATH environment variable, configuring on Windows, 25–26
- path separator character (/ or \), 27
- pdb module, 250–251
- performance maintenance, SQLAlchemy and, 162–163
- permission check, for authorization, 415
- permission objects (AuthKit), @authorize decorator and, 421–422
- PermissionError
 - @authorize decorator and, 421
 - ways of checking permissions that raise, 422
- permissions, changing templates based on, 438
- pickle module (Python), 131
- PickleType field (SQLAlchemy), 139
- Pilgrim, Mark, 8
- pipelines, 396–397
- pixels to percent translation for fonts, 327
- pkg_resources package, 392
- pkg_resources.ExtractionError error, 21
- plural forms and internationalization, 241–242
- pool_recycle option (MySQL), 171
- populating one drop-down list from values of another, 356–360
- port 443, 431
- portable object (.po) file, 229
- portable object template (.pot) file, 229
- POST method, 40, 93–95
- posting traceback information online, 58
- prevalidators, 119
- primary key, 146
- private data, 415–416
- private key for SSL, 429
- private members (JavaScript), 346
- private() action, 420
- production config file template, customizing, 457–459
- production configuration file, 485
- production environments, interactive debugger in, 59
- project
 - application directory structure, 37–38
 - creating, 31–32
 - directory structure of, 36–37
 - running with web server
 - configuration files and, 33–34
 - IP addresses, hosts, and security, 36
 - Paste HTTP server, 34–35
 - static files, 35–36
- project template, 32
- propagation
 - capturing SQLAlchemy log messages using, 483–484
 - controlling with loggers
 - filtering messages, 482
 - options, 483
 - overview of, 480–482
 - definition of, 473

- protecting
 - controllers, 424
 - delete() action, 437
 - edit() action, 436
 - save() action, 436
 - prototypal inheritance, 339, 347–348
 - prototypes (JavaScript), 348–349
 - proxy server, installing with, 20
 - proxying approach to deployment
 - Apache and
 - init scripts, creating, 496–497
 - log files, setting up, 496
 - overview of, 494–495
 - restarting stopped applications, 497
 - overview of, 494
 - public directory, 38, 325
 - public folder, 36
 - public/css/main.css, style, adding, 321–322
 - Pygments package, 277
 - Pylons
 - community support, 9
 - components, 9
 - convention over configuration, 6
 - features of, 7–8
 - installing, 17
 - loose coupling and clean separation, 7
 - Model View Controller architecture of, 5–6
 - overview of, 3, 11
 - techniques of, 5
 - pylons project template, 32
 - Pylons-0.9.7-py2.5.egg package, 30
 - PylonsApp instance, 406, 410–411
 - PylonsInstaller object, 392
 - pylons_minimal template, 32
 - PyPI. *See* Python Package Index
 - pysqlite2 module, installing, 135
 - Python 2, Unicode in
 - decoding, 221
 - encoding, 222
 - handling errors, 220–221
 - literals, 219–220
 - overview of, 219
 - source code encoding, 222–223
 - writing data to files, 223
 - Python CGI script, 4–5
 - python command, 23
 - Python language
 - See also* Python 2, Unicode in
 - blocks, code within, 68–69
 - C and C++ extensions, support for, 23
 - compiling directly from source, 23
 - debugger (pdb), 250
 - eggs, 18
 - IDLE editor, 28
 - as interpreted, 13
 - object-relational mappers available for, 133
 - operators, 144
 - overview of, 8
 - pickle module, 131
 - source code, documenting with Sphinx, 273–276
 - versions of, 8, 17, 22
 - Python Package Index (PyPI)
 - DB-API drivers and, 134
 - downloading from, 14
 - publishing egg on, 460–462
 - SimpleSite on, 457–461
 - python script, 31
 - PYTHON_EGG_CACHE environment variable, 21
- ## Q
- queries (SQLAlchemy), 157–159
 - querying data, SimpleSite example, 175–176
 - quick installation on Linux, 13–14
- ## R
- RDBMS (relational database management systems)
 - See also* SQLAlchemy
 - DB-API drivers for, 133–134
 - object databases compared to, 131
 - object-relational mappers, 132–133
 - README.txt file, 37
 - redirecting log output using handlers
 - logging to files, 478
 - logging to wsgi.errors, 479–480
 - overview of, 477
 - redirect_to object, 47
 - redirect_to() function, 202–203, 412
 - reducing page load time, 365–366
 - Refresh button and resubmitted data problem, 95–96
 - register command, 460
 - RegistryManager component (WSGI), 404, 409
 - relation() function, 151
 - relational database, and object-relational principles, 146–148
 - relational database management systems. *See* RDBMS
 - reload option (Paste HTTP server), 55
 - reload switch, starting server with, 166
 - RemoteUser permission, 421
 - removing
 - DOM nodes, 353
 - files before packaging project for distribution, 459
 - packages, 20
 - render() function
 - assigning template variables to c compared to passing them directly as arguments to, 65
 - HTML Fill tool, 110–111
 - linking to template engine code, 85–86
 - objects passed automatically via, 69–70
 - render_body() function, 78
 - render_mako() function, 86

- render_signin() function, 442
- render_template() function, 86
- repeating fields problem on forms
 - complete code for, 122–124
 - controller code for, 121
 - field names for, 119
 - overview of, 115
 - role field values, 116
 - schema for, 116–121
 - solutions to, 116
 - testing, 124–125
- repoze.who toolset, 418
- _repr_ method, 150
- request cycle, Unicode and, 226
- request handling
 - Cascade, 408
 - middleware chain, 410
 - overview of, 407–408
 - PylonsApp instance, 410–411
 - RegistryManager, 409
 - WSGIController, 411
- request information, Pylons compared to WSGI, 370
- request object
 - environment variables and, 42
 - methods and attributes of, 44–45
- request parameters, decoding, 224
- request state, 404
- request.environ dictionary, 43
- request.params object, 45, 93
- request.params.getall() method, 45
- request.params.getone() method, 45
- request.urlvars dictionary, 199, 440
- requests
 - Ajax, debugging, 360–361
 - environment variables set for all, 42
 - HTTP, 40
- requirement argument (Routes), 211
- resetting browser CSS, 326–327
- response handling
 - error documents, 412
 - exception handling, 412
 - overview of, 411–412
 - returning Unicode from action, 413
 - streaming content, 412–413
- response object, 44–46
- responses
 - HTTP, 40
 - Pylons compared to WSGI, 370
 - returned from WSGI application, altering, 383–387
- restarting stopped applications, 497
- @restrict decorator, 181
- restricting access using _before_() method, 416–417
- reStructuredText language, 268–270
- resubmitted data problem, 95–96
- ResultProxy object (SQLAlchemy), 136, 144
- results, selecting (SQLAlchemy), 143–146
- retry timeout, 495
- returning Unicode from action, 413
- \r\n (line-end) character (Windows), 27
- roles, authorization and, 424–426
- ROLLBACK statement (SQLAlchemy), 155
- rolling back (SQLAlchemy), 152
- root logger, 473
- root URL, mapping to controller action, 39
- route map, 196–198
- route memory feature, 207–208
- route minimization feature, 207
- route options, 196
- route paths, 196
- routes
 - for comments system, modifying, 280–281
 - definition of, 196
 - dynamic parts of, 195
 - explicit, 202, 210
 - generating, functions for, 202–203
 - internal static, 204
 - named
 - description of, 203–204
 - filter functions and, 215–216
 - naming, 195
 - parts of, 199–201
 - static named, 204
- Routes system
 - best practice, 209–211
 - c object and, 49
 - conditions argument, 212–215
 - controller_scan() function, 198
 - default variables, specifying, 201–202
 - description of, 30, 195
 - filter functions, 215–216
 - Mapper object, 197
 - modifying, 38–39
 - requirement argument, 211
 - specifying extra variables in route maps, 48
 - unnecessary features of
 - implicit defaults, 209
 - overview, 196, 207
 - route memory, 207–208
 - route minimization, 207
 - url_for() function, generating URLs with, 195
- RoutesMiddleware component (WSGI), 404–405, 410
- routing for SimpleSite, changing, 313–319
- routing map, and connect() function, 201
- routing variables
 - calling controller action with, 198–199
 - definition of, 196
 - naming, 202
- routing.py file, 51
- Ruby on Rails, and clean separation, 7
- runtime built-ins (Mako), 70–72

S

- same origin policy, Event Model, 354
- save() action
 - protecting, 436
 - testing, 257–260
- save() method, updating controller to support
 - editing pages, 183–184
- SciTE editor, 28
- scoped_session() function, 153
- scripts, 31
- Scripts directory, 31
- search engines
 - internationalization and, 243
 - use of, study of, 205
- search locations for nose, 251
- secondary table, specifying, 151
- section links for SimpleSite, 319–321
- Secure Sockets Layer (SSL), setting up, 429–431
- security
 - authentication
 - encrypting passwords, 431–432
 - overview of, 429
 - SSL, 429–431
 - interactive debugger in production
 - environments, 59
 - naming files and, 100
 - running project with web server, 36
 - view templates and
 - applying filters, 75–76
 - overview of, 73–74
 - writing helpers to use HTML literals, 74–75
- SELECT statement (SQLAlchemy), 143
- select() helper, 97–98
- selecting results (SQLAlchemy), 143–146
- self.app object, 252
- Sequence object (SQLAlchemy), 149
- server component (WSGI), 374–378
- server logs
 - application logs compared to, 471–472
 - sending messages to wsgi.errors stream and, 479
- servers
 - See also* web server, running project with
 - constructing, Paste Deploy and, 393–394
 - embedding applications into, 493, 497–502
 - Paste HTTP, 34–36, 55, 430
 - proxy, installing with, 20
 - Pylons, SSL and, 430
 - starting, 166
 - WSGI, accessing programmatically, 401–402
- server_runner() function, 394
- serving application from installed environment, 492
- session handling, Beaker package and, 190
- session object, 48
- session.commit() method, 156
- session.execute() method, 156
- sessionmaker() function, 152
- SessionMiddleware component (WSGI), 404, 410
- sessions
 - Beaker package, 152
 - SQLAlchemy
 - examples of, 153–156
 - overview of, 152–153
 - using in Pylons, 177
 - storing user language in, 238–239
- setup.cfg file, 37
- setup.py file
 - overview of, 37
 - for SimpleSite
 - dependencies, 453–454
 - extra dependencies, 454–455
 - extra dependency links, 455
 - long_description argument, 456–457
 - metadata, specifying, 455–456
 - overview of, 452
 - production config file template, 457–459
 - version number, choosing, 453
- setuptools package
 - description of, 30
 - development mode, 174
 - extra dependency feature, 454–455
 - upgrading, 21
 - version number and, 453
- setup_app() function, 174, 255–256
- set_lang() function, 239
- shutil module, copytree() function, 129
- shutil.copyfileobj, 99
- sign-in screen, styling, 441–442
- sign-in, triggering, 423
- signedin.html template, 439
- signedout.html template, 439
- signinagain() action, 448–449
- signing in and out
 - AuthKit middleware and, 421
 - of PyPI, 461
 - of SimpleSite, 439–440
- signout() action, 420
- simplejson package, 30
- SimpleSite application
 - Ajax and, 356–360
 - animation, adding to, 354–356
 - API Documentation page, 274
 - AuthKit and
 - changing templates based on permissions, 438
 - controller actions, protecting, 436–438, 443–444
 - middleware, setting up, 433–434
 - signing in and out, 439–440
 - styling sign-in screen, 441–442
 - User Management API, 445
 - websetup.py, adjusting, 434–436
 - base controller role, 176–177

- comments system
 - controller, creating, 281–282
 - controller, planning, 280
 - controller, updating to handle comments, 282–285
 - overview of, 279–280
 - routes, modifying, 280–281
 - setting page ID automatically, 285–288
 - controller
 - create() method, 180–183
 - customizing, 167
 - delete() method, 185–186
 - edit and save() methods, 183–184
 - list() method, 185
 - new() method, 178–180
 - updating to support editing pages, 177
 - view() method, 178
 - creating new project, 166–167
 - database tables for, creating, 173–175
 - dates and times, formatting, 189–190
 - deployed with apache and mod_wsgi, 500
 - deploying
 - activate script, 490
 - virtual Python environment and, 489
 - documenting with Sphinx, 270–273
 - error documents for, customizing, 447–449
 - finished, with customized front page, 165
 - flash message, using, 190–192
 - footer, updating, 186
 - functional tests for, 252
 - grids, using, 332–336
 - Index page, 275
 - making into Paster project template
 - overview of, 462–465
 - variables, 465–468
 - model for
 - creating, 171–173
 - engine, configuring, 171
 - overview of, 170
 - navigation elements, adding, 319–321
 - navigation hierarchy
 - controllers, creating, 303–311
 - creating, 299
 - inheritance in SQLAlchemy and, 299–301
 - initial data, setting up, 301–303
 - page controller, 311–313
 - on PyPI, 457, 461
 - overview of, 165
 - packaging for distribution
 - egg file, building, 459–460
 - egg file, publishing in PyPI, 460–462
 - page controller, creating, 167–168
 - Page view
 - deleted pages, handling, 289–290
 - updating, 288–289
 - pages, deleting, 298
 - pagination, using, 186–189
 - querying data, 175–176
 - route map, 196
 - routing, changing, 313–319
 - setup.py file, configuring
 - dependencies, 453–454
 - extra dependencies, 454–455
 - extra dependency links, 455
 - long_description argument, 456–457
 - metadata, specifying, 455–456
 - overview of, 452
 - production config file template, 457–459
 - version number, choosing, 453
 - setup_app() function, 255–256
 - SQLAlchemy sessions, using, 177
 - style, adding, 321–323
 - tags
 - adding to pages, 295–298
 - deleting, 298
 - names, constraining, 293–294
 - overview of, 290
 - tag controller, creating, 291–293
 - template, releasing, 468–469
 - template structure, 168–170
 - WYSIWYG interface, adding, 449–452
- SOLObject, 133
- source code
 - See also* code; listings
 - commenting, 263–264
 - encoding (Unicode), 222–223
 - Python, documenting with Sphinx, 273–276
- special nested grids, 331–332
- Sphinx tool
 - automatically generating documentation using, 276–277
 - docstrings and, 266
 - documenting Python source code using, 273–276
 - documenting SimpleSite project using, 270–273
 - syntax highlighting, 277–278
- sphinx-build command, 271
- SQL (Structured Query Language), RDBMS and, 132
- SQL Expression API (SQLAlchemy)
 - injection attacks and, 142–143
 - overview of, 141–142
 - selecting results, 143–146
- SQL injection attacks, 142–143
- SQLAlchemy
 - See also* navigation hierarchy
 - architecture
 - Declarative API, 160–162
 - Engine API, 136–138
 - Metadata and Type APIs, 138–141
 - overview of, 135–136
 - SQL Expression API, 141–146
 - automatically converting string types to handle Unicode, 140
 - base controller, role of, 176–177

- capturing log messages using propagation, 483–484
 - database tables, creating, 173–175
 - description of, 133
 - engine, configuring, 171
 - inheritance in, 299–301
 - installing, 135
 - maintaining performance, 162–163
 - model, creating, 171–173
 - Object-Relational API
 - classes and mappers, 150–151
 - database metadata, describing, 148–150
 - objects, 159–160
 - overview of, 146
 - queries, 157–159
 - sessions, 152–156
 - object-relational mapper, 6
 - overview of, 170
 - query object, 175–176
 - sessions, using in Pylons, 177
 - setting up, 133
 - Unicode column type, 225
 - SQLAlchemy driver (AuthKit), 434
 - SQLite
 - creating database with, 135
 - description of, 134
 - installing, 134–135
 - memory mode function, 137
 - specifying relative an absolute paths in, 137
 - SSL (Secure Sockets Layer), setting up, 429–431
 - stability of object databases, 131
 - stack, building out of WSGI middleware, 389–390
 - StackedObjectProxy class, 409
 - starting
 - Paste HTTP server, 35
 - server, 166
 - start_response argument, 199, 202
 - start_response() callable
 - custom, to return GzipFile object, 384–385
 - description of, 371
 - final version of code, 386–387
 - updating GzipMiddleware class to call, 385–386
 - WSGI middleware and, 379
 - WSGI servers and, 374
 - static files, 35–36
 - static named routes, 204
 - static parts of routes, 200
 - static-looking URL, 205
 - StaticURLParser, 407
 - status codes (HTTP), 41
 - status in piece of middleware, changing, 381
 - StatusCodeRedirect component (WSGI), 404, 410
 - StatusCodeRedirect middleware
 - 403 error document and, 437
 - error documents and, 445–446
 - storing
 - user language in sessions, 238–239
 - view templates, 64
 - storing data
 - See also* RDBMS
 - in Amazon S3, 129–130
 - approaches to, 127
 - in databases
 - object databases, 131
 - overview of, 130
 - XML databases, 131–132
 - in filesystems, 127–129
 - Storm, 133
 - story object, filter function for, 215–216
 - streaming content, 412–413
 - strftime() method (datetime.datetime objects), 189–190
 - strict_c option, 50
 - String data type (JavaScript), 341
 - string types, automatically converting to handle Unicode, 140
 - strings
 - concatenating, 145
 - extracting, and handling translations, 229
 - marking for internationalization, 228–229
 - Structured Query Language (SQL), RDBMS and, 132
 - style for SimpleSite, adding, 321–323
 - stylesheet_link() helper, 182
 - styling sign-in screen, 441–442
 - submit() helper, 96
 - submitting forms using GET or POST HTTP methods, 93–95
 - sub_domain option (conditions argument), 213–215
 - Supervisor tool, 497
 - supporting multiple languages, 232–234
 - syntax
 - highlighting (Sphinx), 277–278
 - template, 66–69
 - system Python environment, 488
- ## T
- table columns, naming, 151
 - tables
 - for laying out HTML content, 328
 - for SimpleSite, creating, 173–175
 - in SQLAlchemy, 138–140
 - tag controller, creating, 291–293
 - Tag object (SQLAlchemy), 151
 - tags for SimpleSite
 - adding to pages, 295–298
 - deleting, 298
 - names, constraining, 293–294
 - overview of, 290
 - tag controller, creating, 291–293
 - temp file data, copying to permanent location, 99

- Tempita template language, 30
- template preset grids, 329–331
- template structure for SimpleSite, 168–170
- Template tab (interactive debugger), 56
- template variables, default, 69–70
- templates
 - See also* Paster project template; *specific templates*; view template
 - changing based on permissions, 438
 - derived/form.html, 117
 - for forms, 92
 - listing, 32
 - logging module, 474–475
 - .pager(), variables to use as arguments in, 188
 - separating from models, 176
 - translations within, 235–236
 - types of, 32
- templates directory, 38, 64
- templates/base/index.html template, 321
- templates/component/navigation.html template, 320–321
- templates/derived/comment/fields.html template, 283
- templates/derived/page/view.html template
 - footer def, 289
 - page controller, 296
- templating, Unicode and, 225
- templating languages
 - See also* Mako templating language
 - choosing among, 88
 - integrating into Pylons application, 90
 - Jinja 1 and Genshi, 88–89
 - working with multiple, 89–90
- templating system, 63. *See also* view template
- test.ini file, 254–256
- testing
 - form to handle repeating fields problem, 124–125
 - from command line, 261–262
 - functional
 - for controllers, 428
 - description of, 246
 - of objects, 260–261
 - page controller save() action, 257–260
 - with paste.fixture, 252–256, 261
 - JavaScript in Firebug, 337
 - middleware, 387–388
 - overview of, 245–246
 - POST method, 95
 - types of, 246–247
 - unit, with nose, 247–251
 - view templates, 64
- tests directory, 38
- test_page object (SQLAlchemy), 153–156
- test_save() method, 259–260
- test_save_invalid_form_data() method, 259
- test_save_invalid_id() method, 258
- test_save_prohibit_get() function, 258
- text() helper, 96–97
- <%text> tag, 67
- Thawte certificate authority, 429
- this (JavaScript), 345–346
- 301 redirect to URL, 36
- time.sleep() function, 413
- times for SimpleSite, formatting, 189–190
- time_ago_in_words() function, 129
- tinyurl.com, 205
- tmpl_context object, 49–50
- toctree directive (Sphinx), 273
- tools
 - See also* documentation tools; FormEncode tool; HTML Fill tool; nose tool
 - Buildout, 15, 488–489
 - for creating virtual Python environments, 15
 - decorator, 29
 - editors, 28
 - Firebug
 - console object, 362
 - description of, 336
 - Inspect button, 351
 - Net tab, 360
 - testing JavaScript in, 337
 - or form handling, 91
 - mod_wsgi
 - embedding with, 497–499
 - overview of, 493
 - troubleshooting, 501–502
 - virtual host, setting up, 499–500
 - to monitor and restart processes, 497
 - navigator, 187
 - repoze.who
 - SciTE editor, 28
 - Sphinx
 - automatically generating documentation using, 276–277
 - docstrings and, 266
 - documenting Python source code using, 273–276
 - documenting SimpleSite project using, 270–273
 - syntax highlighting, 277–278
 - Supervisor, 497
 - ToscaWidgets, 91
 - virtualenv.py, 15, 19
 - ToscaWidgets tool, 91
- to_python() method, 105
- Traceback tab (interactive debugger), 56
- TranslateDemo example
 - Babel, using, 230–232
 - overview of, 229–230
 - supporting multiple languages, 232–234
 - translations within templates, 235–236
 - updating message catalogs, 234–235

- translations for internationalization
 - handling, 229
 - lazy, 240–241
 - within templates, 235–236
- trickling-down model of event handling, 352
- troubleshooting
 - Easy Install, 21–22
 - mod_wsgi, 501–502
- trove classifiers, 455
- TurboGears 1.0, 22
- turning off debug mode in INI file, 412
- Type API (SQLAlchemy), 138–141
- types (JavaScript), 341–342

U

- U flag for commands, 21
- UI, treating URL as part of, 206
- UKDateConverter validator, 108
- Undefined data type (JavaScript), 341
- underscore (`_`) character, 66, 415
- ungettext() function, 241
- unichr() function, 220
- Unicode
 - automatically converting string types to
 - handle, 140
 - complete request cycle, 226
 - description of, 218–219
 - encoding code points into binary numbers, 218
 - error message, 217
 - history of, 217–218
 - in Pylons application, 224–225
 - in Python 2
 - decoding, 221
 - encoding, 222
 - handling errors, 220–221
 - literals, 219–220
 - overview of, 219
 - source code encoding, 222–223
 - writing data to files, 223
 - returning from action, 413
 - UTF-8 encoding and, 219
- Unicode() constructor, 220–221
- unicodedata module, 220
- UniqueSectionPath validator, 305
- UniqueTag validator, 293
- unit testing
 - description of, 246
 - with nose
 - command-line debugging, 250–251
 - debug messages, 249
 - detailed errors, 249
 - overview of, 247–249
 - search locations, 251
- UPDATE statement (SQLAlchemy), 145
- update_tags() action, 297

- updating
 - controller
 - create() method, 180–183
 - delete() method, 185–186
 - edit and save() methods, 183–184
 - list() method, 185
 - new() method, 178–180
 - to support editing pages, 177
 - view() method, 178
 - footer for SimpleSite, 186
 - message catalogs, 234–235
- upgrading
 - packages with Easy Install, 20
 - setuptools, 21
- uploading files, 98–101
- URL
 - choosing, tips for, 205–206
 - generating, 195
 - mapping, 195
 - matching, 197
 - parts of, 204
- url_for() helper, 96
- User Management API (AuthKit)
 - overview of, 426
 - SimpleSite and, 445
- user testing
 - description of, 246
 - resources on, 247
- UserIn permission, 422
- UsersFromFile driver (AuthKit), 426
- UsersFromString driver (AuthKit), 426
- UTF-8 encoding, 218–219

V

- validate() decorator, 103, 111–112
- validate_python() method, 113
- validating e-mail address, 101–102
- validation schema for forms, 103
- validators
 - accessing objects that aren't thread-safe, 294
 - chained, 296
 - description of, 103
 - FormEncode tool
 - chained, 121
 - configuring, 107–108
 - custom, creating, 112–115
 - list of, 106–107
 - options supported by, 107
 - prevalidators, 119
 - ValidAuthKitUser permission, 422, 436–437
 - ValidSectionPosition validator, 305
 - ValidTagsForm schema, 297
- variables
 - assigning template, using context c global, 65–66
 - cache_dir, 78
 - century routing, 201
 - default (Routes), 201–202

- defining, in JavaScript, 339
 - environment, 42–44
 - hard-coded (Routes), 201
 - HTTP_PROXY environment, 20
 - logging methods, 474
 - project template, 465–468
 - PATH environment, 25–26
 - PYTHON_EGG_CACHE environment, 21
 - routing
 - calling controller action with, 198–199
 - definition of, 196
 - naming, 202
 - WSGI, 374
 - wsgi.errors, 375
 - VeriSign certificate authority, 429
 - version number for project, choosing, 453
 - view template
 - applying filters in, 75–76
 - assigning variables using context c global, 65–66
 - basic syntax, 66–69
 - body() def, 81
 - caching functionality (Beaker package), 87
 - capturing output, 79–80
 - def blocks, using, 76–77
 - description of, 6, 32, 64
 - inheritance chains
 - next namespace, 83–85
 - overview of, 81
 - parent namespace, 85
 - simple inheritance, 82–83
 - linking render() function to engine code, 85–86
 - Mako caching, 77–79
 - Mako runtime built-ins, 70–72
 - <%namespace> tag, 80–81
 - objects passed automatically via render() function, 69–70
 - security considerations, 73–74
 - separating logic code and view code, 72
 - storing, 64
 - testing, 64
 - writing helpers to use HTML literals, 74–75
 - view() method
 - page controller, 295
 - updating controller to support editing pages, 178
 - view.html template
 - for SimpleSite, 170
 - tag controller and, 292
 - updating to handle comments, 283
 - viewing environment variables, 43–44
 - virtual host, mod_wsgi and, setting up, 499–500
 - virtual Python environment
 - activate script, 490
 - activating and deactivating, 18–19
 - application instance, setting up, 491
 - Buildout, 488–489
 - config file for application, creating, 491
 - installing required software to, 490–491
 - serving application, 492
 - setting up, 14–15, 489
 - virtualenv.py tool
 - creating virtual Python environment with, 15
 - setting options for, 19
- ## W
- Web Developer toolbar, 40
 - web server, running project with
 - configuration files and, 33–34
 - IP addresses, hosts, and security, 36
 - Paste HTTP server, 34–35
 - static files, 35–36
 - Web Server Gateway Interface. *See* WSGI
 - web site example. *See* SimpleSite application
 - web sites
 - community support, 9
 - doctest module documentation, 268
 - Easy Install documentation, 16
 - ECMAScript specification, 340
 - on filesystem use, 129
 - Firebug, 336–337
 - Firefox web browser, 40
 - GNU gettext, 229
 - HTML helpers documentation, 97
 - HTTP specification, 41
 - listamatic, 322
 - Mako documentation, 70
 - Mercurial documentation, 22
 - object databases, 131
 - object-relational mappers, 133
 - Paste Script developer documentation, 393
 - paste.fixture documentation, 261
 - Python Package Index, 14
 - Python Tutorial, 8
 - reStructuredText documentation, 270
 - Sphinx documentation, 278
 - Web Developer toolbar, 40
 - WebHelpers documentation, 75
 - XML databases, 132
 - YUI, 326
 - WebError package, 30
 - WebHelpers package
 - building forms with, 96–98
 - description of, 31
 - HTML helper functions, writing or upgrading, 74–75
 - paginate module, 187
 - version 0.6, changes in, 96
 - webhelpers.html.escape function, 73
 - webhelpers.html.literal() object, 73
 - WebOb package
 - description of, 31
 - request object and, 44
 - response object and, 46

- websetup.py file
 - description of, 38
 - egg entry points and, 391–392
 - for navigation, updating, 301–303
 - updating for SimpleSite, 434–436
- WHERE clause (SQLAlchemy), 143–144
- whitespace, help() function and, 265
- wiki comments system as object-relational
 - example, 147–148
- wildcard parts of routes, 200–201
- Windows
 - deployment on, 502
 - developing applications on, 27–28
 - file extensions for commands, 16
 - installation on, 24–26
 - Python versions and, 17
- writing
 - helper functions to use HTML literals, 74–75
- WSGI middleware
 - environment, modifying, 380–381
 - errors, handling, 381–382
 - overview of, 379–380
 - response, altering, 383–387
 - status and headers, changing, 381
 - testing, 387–388
 - Unicode data to files, 223
- WSGI (Web Server Gateway Interface)
 - application component, accessing
 - programmatically, 401–402
 - applications
 - as class instances, 371–372
 - overview of, 370–371
 - as Pylons controllers, 372–374
 - architecture, history of, 389–390
 - components of, 369
 - composite applications, 395–396
 - description of, 43, 369
 - middleware
 - advantages of, 389
 - building stack out of, 389–390
 - overview of, 378–379
 - testing, 387–388
 - writing, 379–387
 - servers, 374–378
 - variables, 43

- wsgi.errors, logging to, 479–480
- wsgi.errors variable, 375
- WSGIController, 411
- WSGIErrorsHandler, 479–480
- wsgiref module, 378
- WYSIWYG interface, adding to SimpleSite, 449–452

X

- X-Debug-URL header, 58
- XML databases, 131–132
- XSS (cross-site scripting) attacks, 73

Y

- YAHOO namespace (YUI library), 346
- YAHOO.lang, functions defined in, 341–342
- YUI
 - adding to project, 325–326
 - animation classes, 354
 - event handling in, 353
 - fonts style sheet, 327–328
 - grids
 - nested, 331
 - special nested, 331–332
 - style sheet, 328–329
 - template preset, 329–331
 - updating SimpleSite to use, 332–336
 - library
 - YAHOO namespace, 346
 - YAHOO.lang, functions defined in, 341–342
 - resetting browser CSS with, 326–327
- YUI Rich Text Editor
 - preventing HTML page content from being
 - escaped, 452
 - setting up, 449–450
 - styles and, 452
 - theming system, 450–452
- yui-gf class, 331

Z

- 0.0.0.0 IP address, 36
- ZIP files, 18
- ZODB object database, 131