

Index

- \$ (dollar sign) in Python expressions, 115**
- % (percent) for embedded tags, 87**
- @login_required decorator (Django), 293**
- { } (braces) for embedded tags, 87**
- in Python expressions, 115**
- {% extends %} directive (Django), 279–282**

- A**
- Academic Free License (AFL). See AFL (Academic Free License)**
- ActionScript**
 - binding expressions in, 390
 - vs. ECMAScript, 365–366
 - in Flex-based RIAs, 364–365
 - Logic Moved to (Flex Code Tester), 373–375
 - for user interaction (Flex), 362
- addjob controller method (Cable Company), 219–221**
- admin interface, Django, 251–254**
- administration utility (Django)**
 - adding models to, 81–82
 - adding utility, 79–81
 - applying, 82–84
 - tailoring, 84–85
- AFL (Academic Free License), 14**
- AJAX (Asynchronous JavaScript and XML)**
 - AJAX -ready blog_detail.html, 313–314
 - blog application. See application examples (Ajax blog)
 - Django serialization for, 307–311
 - Django support for, 306
 - role in Web 2.0, 10
- all method (Django), 79**
- alternate components**
 - Django, 72
 - TurboGears, 52–53
- Apache HTTPD server, installing, 327–329**
- application design, history of**
 - desktop era, 4–5
 - Web 1.0, 5–6
 - Web 2.0, 6–10
- application examples (Ajax blog)**
 - applying AJAX effects, 311–312
 - blog_detail.html template, 313–314
 - Django view and getComments function, 318–320
 - JavaScript in dblog.js, 314–318
- application examples (Cable Company)**
 - addjob controller method, 219–221
 - assignment model class, 215–216
 - complete controller, 222–223
 - controller error handling, 223–227
 - imports class, 216–217
 - job model class, 213–214
 - location class defined, 211–213
 - master.kid template, 228–229
 - model objects, 210–211
 - operational concept, 205
 - overview, 204
 - priority model class, 213–214
 - resource model class, 215
 - resourcelist method, 218–219
 - root class, 217–218
 - RSS feed capability, 234–237

application examples (continued)

application examples (continued)

- status model class, 215
- view, displaying map data, 232–233
- view, displaying table data, 233–234
- view, getting data, 232
- view, map loading, 230–231
- view, setting up and debugging, 230–231

application examples (Dblog)

- {% extends %} directive, 279–282
- admin interface, editing model with, 251–254
- authenticating users, 287–288
- base template, defining, 277–279
- base template, enhancing, 290–291
- blog detail with inheritance, 280–281
- comment model classes, creating, 249–251
- comments, entering, 269–273
- create_update objects generic views, 285
- date-based generic views, 284
- drilling down on blog object, 264–266
- form processing, 266–269
- index view method, 255–259
- initializing application, 245
- initiating with django-admin.py, 242–245
- list-detail generic views, 282–284
- master list template without inheritance, 276–277
- model classes, creating, 246–249
- model with custom permission, 294–295
- new user entry form template, 301–302
- permissions on submitted data, 296–297
- simple.direct_to_template generic view, 284
- simple.redirect_to generic view, 284
- URL mapping, 260–264

- user login and profile, 288–290
- user management, installing, 285–286
- user model, 286–287
- user permissions, 293–297
- user profile template and view, 290
- users, creating, 297–302
- view for adding user account, 299–301
- view requiring custom permission, 295–296
- view with login required, 294

application examples (Django)

- administration application, 81–85
- administration utility, 79–81
- creating project skeleton, 73–75
- database creation, 76–77
- database initialization with interactive shell, 77–79
- defining model classes, 76
- templates, 86–89
- view, adding to URL list, 89–90
- view, defining, 90–92

application examples (FBlog)

- application controller, 382–384
- connecting to TurboGears controller, 390
- creating states, 389–390
- deconstructing, 387–392
- deploying, 391–392
- event handlers, 388–389
- Flex Eclipse plug-in, 388
- form handling, 391
- illustrated, 380–381
- MXML implementation, 385–387
- supporting model, 381–382

application examples (Flex Code Tester)

- adding effects, 377–378
- all logic in MXML, 366–372
- attribute customization, 376–380
- logic moved to ActionScript, 373–375

application examples (Hello Widget)

- adding template path to controller, 186–188

 application examples (TurboGears)

- widget, defining, 185–186
- widget in template, 186–187
- application examples (Pets)**
 - architecture, 29
 - button handling functions (JavaScript), 34
 - CGI program, 42–45
 - database, defining Python class, 38–39
 - database, initializing, 39–40
 - database, restoring/retrieving objects, 40–42
 - HTML page structure, 30–32
 - queryHandler function (JavaScript), 35–36
 - utility functions (JavaScript), 33
 - workings of, 25–28
 - XML error response (JavaScript), 37–38
 - XML parsing functions (JavaScript), 36–37
 - XML response (JavaScript), 37
- application examples (RSS)**
 - accessing RSS feed, 325–327
 - blog feed example, 321–322
 - creating RSS feed, 323
 - LatestEntries feed class, 323–325
- application examples (Tblog)**
 - creating database for model, 107–108
 - creating project with tg-admin, 105
 - creating users in CatWalk, 145
 - defining model classes, 105–107
 - displaying author name, 149–150
 - entries list, adding return link to, 157–159
 - identity module, 150–151
 - master template, 132–136
 - page access, restricting in controller, 151–154
 - password encryption, 145–148
 - removing BlogAuthor model and controller, 148
 - restricting element display in template, 156–157
 - selecting author based on logged-in user, 154–156
 - statistics page, adding link to, 166–168
 - style sheet, defining, 134–135
 - TGCrud, creating blog author views with, 110–118
 - TGCrud, creating blog entry views with, 118–127
 - updating BlogEntryController, 148–149
 - updating model to User SQLAlchemy, 145
 - visitor tracking model object, defining, 162–163
 - visitor tracking plug-ins, 163–164
 - visitor tracking template, 164–166
 - welcome page, 128–136
- application examples (Tblog User Comments)**
 - accepting User Comments, 174–177
 - BlogComment model object, 173–174
 - comment widget. See comment widget
 - form widget, defining, 175–177
 - Kid template, creating, 177–178
 - linking to new comments page, 182–184
 - save operation, defining, 180–182
 - template, incorporating in controller, 178–180
- application examples (TurboGears)**
 - CatWalk, adding to controller, 63–64
 - controller, calling view from, 66–67
 - controller, defining, 61–62
 - creating with tg-admin tool, 54–57
 - database creation/initialization, 58–61
 - extending Kid template to display names, 68–69
 - model classes, defining, 57–58
 - model object editor, 62–63

application examples (continued)

application examples (continued)

search view, defining in controller, 69–70

tailored view, defining, 64–66

arrays, associative (Python), 350–351

assignment model class (Cable Company), 215–216

associative arrays (Python), 350–351

Async module, MochiKit, 342–349

asynchronous external method calls, 10

Asynchronous JavaScript and XML (AJAX). See AJAX (Asynchronous JavaScript and XML)

asynchronous outbound calls, 8

Atom, 321

attributes

Python, 115–116

SQLObject, 106–107

widgets, 111, 186, 192

authentication, user (Dblog), 287–290

authorization, user (Dblog), 285

automated ORM, 51

B

Base module, MochiKit, 349–351

binding expressions (ActionScript), 390

blog applications. See application examples

browser, widget, 172

button handling functions (Pets), 34

C

call restrictions, 345–349

Canvas container (Flex), 364

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

CatWalk utility (TurboGears)

accessing, 108

adding to controllers, 63–64

creating users in (Tblog), 145

embedding in applications, 62–63

encrypted passwords and, 147

fundamentals, 209–210

initializing database with, 60–61

CGI (Common Gateway Interface)

basic use of, 25

program (Pets application), 42–45

CherryPy framework (TurboGears)

defined, 53

documentation for, 102

cleanupResponseString() function, 33

clearResults() function, 33

code listings

addComment template (Dblog), 270–271

AJAX -ready blog_detail.html, 313–314

Base template, 278–279

Base template with login/logout functions, 291–293

blog and comment model objects, 308–309

blog detail template (Dblog), 262–263

Blog detail view, 282

Blog detail with inheritance, 280–281

blog entry template (Dblog), 267–268

blog entry view method (Dblog), 267

blog model specification, 105–106

Blog model with custom permission, 294–295

blog submission view (Dblog), 269

BlogEntryController edit template, 125–127

BlogEntryController list template, 123–124

BlogEntryFields class defined, 119–122

button handling functions (Pets), 34

CGI program (Pets), 42–45

Checking permissions on submitted data, 296–297

- comment entry view method (Dblog), 269–270
- comment list view method (Dblog), 264–265
- comment template (Dblog), 265–266
- comment widget directory structure, 190–191
- comment widget template file, 193
- CommentFields and CommentForm classes defined, 175–176
- comment.js JavaScript file, 194–195
- comments Kid template, 177–178
- controller for BlogAuthors, defining, 111–113
- controllers.py application controller (Cable Company), 222–223
- database, defining Python class, 38–39
- database, initializing (Pets), 39–40
- database, restoring/retrieving objects (Pets), 40–42
- Dblog model classes, 247–249
- dblog.js, 315–317
- Django database creation with syncdb command, 77
- Django HTML template, defining, 88–89
- Django models.py file, 76
- Django shell interaction, 77–79
- Django views.py, 91
- ECMAScript vs. ActionScript, 365–366
- encryption of user passwords, 147–148
- FBlog controllers.py, 382–384
- FBlog HTML file, 392
- FBlog model, 381–382
- FBlog MXML implementation, 386–387
- Feed.py class (Cable Company), 235–237
- Flex code tester-All Logic in MXML, 366–372
- Flex code tester-Attribute Customization, 376–380
- Flex code tester-Logic Moved to ActionScript, 373–375
- HelloWidget specification, 185–186
- helloWorld.kid template, 187
- index template (Dblog), 255–256
- index view method (Dblog), 255
- JSON blog object, 310
- JSON comment data, 318
- Kid template and MochiKit Library, 68–69
- Kid template, defining, 128–129
- Kid template initial version, 64–66
- list template for BlogAuthorController, 114–116
- Master list template with inheritance, 280
- Master list template without inheritance, 276–277
- master template for TBlog, 132–133
- mod_python apache configuration, 328
- New user entry form template, 301–302
- postComment view method (Dblog), 272–273
- Python HTML page generator, 6
- queryHandler function (Pets), 35–36
- readBlog view method (Dblog), 261–262
- Refactored index method (Cable Company), 226–227
- root controller, defining paths in, 67
- root controller extended for beView, 129–131
- RSS feed, 322
- RSS feed class, 323–325
- Show and hide effects method (controlpanel.kid), 234
- show template for BlogAuthorController, 117–118

code listings (continued)

code listings (continued)

- showJobs/showTrucks methods (controlpanel.kid), 232–233
- style sheet for TBlog, 134–135
- Truck method (Cable Company), 223–226
- Truck.kid template (Cable Company), 224
- TurboGears JSON, 218–219
- TurboGears model, 57–58
- TurboGears model.py (Cable Company), 210–211
- TurboGears Python shell interaction, 59
- User login template, 288–289
- User profile template, 290
- User profile view, 290
- utility functions (Pets), 33
- View requiring custom permission, 295–296
- View that adds user account, 299–301
- View with login required, 294
- visitor statistics view, 165–166
- VisitorCountPlugin class defined, 163–164
- widget stylesheet, 196
- XHTML page (Pets), 31–32
- XML blog object, 310
- XML error response (Pets), 37–38
- XML parsing functions (Pets), 36–37
- XML response (Pets), 37
- XMLQueryExample.html, 339–340
- YAML blog object, 311

code tester examples (Flex). See application examples (Flex Code Tester)

CogBin (widgets), 104, 171

comment widget

- appearance, 196
- creating class, 191–192
- getComments path, defining, 197
- packaging, 190–191

- template, defining, 192–193
- using in views, 197–201
- writing JavaScript for, 193–196

comments

- CommentFields object, 176
- CommentForm object, 176
- comments, entering blog (Dblog), 269–273
- comments contrib add-on (Django), 320
- linking to new page, 182–184

Common Gateway Interface (CGI). See CGI (Common Gateway Interface)

community tools, 52

compatibility mode (MochiKit), 357

components

- defined by MVC pattern, 48–49
- Django framework, 71–72
- Identity framework, 144
- TurboGears, 52–53, 101–104

configuration options

- Identity framework, 143
- Visit Tracking framework, 161

containers, user interface (Flex), 364

contrib add-ons, Django, 320–321

contrib.syndication.feeds, Django, 321–323

controllers

- adding CatWalk to (TurboGears), 63–64
- adding template path to, 186–188
- Cable Company application, 222–227
- calling views from (TurboGears), 66–67
- CherryPy in TurboGears, 102
- connecting to TurboGears (FBlog), 390
- controller.py file invocation, 14–17
- creating/using new (TurboGears), 103
- defining (Tblog), 111–113
- defining (TurboGears), 61–62
- defining search view in (TurboGears), 69–70
- FBlog application controller, 382–384
- fundamentals of, 48–51

- Identity framework components
 - and, 144
 - in MVC pattern, 205
 - restricting page access in, 151–152
 - root controller (TurboGears), 102
 - updating BlogEntryController, 148–149
 - controls, user interface (Flex), 363**
 - cookies, 138–140**
 - count function, 130**
 - create_update objects generic views, 285**
 - cross-domain calls, 391**
 - CSS (Cascading Style Sheets)**
 - added to Flex tester example, 371–372
 - css attribute (widgets), 186, 192
 - custom add-ons, Django, 321**
- D**
- Dangoor, Kevin**
 - interview, 93–97
 - and TurboGears creation, 52
 - data validation framework, 50**
 - databases**
 - abstracting interactions, 51
 - creating (Django), 76–77
 - creating for model (Tblog), 107–108
 - creating tables (TurboGears), 58
 - databasetemplateloader add-on (Django), 321
 - defining Python class (Pets), 38–39
 - framework capabilities for developing, 51
 - initialization with CatWalk (TurboGears), 60–61
 - initialization with interactive shell (Django), 77–79
 - initialization with interactive shell (TurboGears), 58–59
 - initializing (Pets), 39–40
 - restoring/retrieving objects (Pets), 40–42
 - databrowse contrib add-on (Django), 320**
 - date-based generic views, 284**
 - Dblog application. See application examples (Dblog)**
 - debugging view (Cable Company), 230–231**
 - decorator function**
 - defined, 11, 102
 - in Python 2.4, 217–218
 - Deferred objects, method signatures on, 344–345**
 - design patterns, for Web 2.0 applications, 15–17**
 - desktop application model, 4–5**
 - desktop publishing (DTP), 5**
 - display method arguments, 178**
 - Django framework**
 - components of, 71–72
 - custom add-ons, 321
 - Dblog application example. See application examples (Dblog)
 - deploying with Apache HTTPD, 327–329
 - design philosophy, 241–242
 - Django application. See application examples (Django)
 - django-admin.py command, 73–74, 242–245
 - django.contrib add-ons, 320–321
 - django.db.models package, 249–251
 - DRY (Don't Repeat Yourself) principle, 276–277
 - forms library, 285
 - history of, 71
 - licensing of, 14
 - MVC architecture in, 72–73
 - overview of, 13–14
 - serialization for AJAX, 307–311
 - tags available in, 88

Django framework (continued)

- template filters, 257–258
- template inheritance, 276–282
- template tags, 256–257
- vs. TurboGears, 11–12
- user authentication/authorization.
See user authentication (Dblog)

Document Object Model (DOM). See **DOM (Document Object Model)**

Dojo Toolkit, 53, 341

DOM (Document Object Model)

- MochiKit DOM API, 195
- MochiKit.DOM module, 351–352

Domain-specific languages (DSLs). See **DSLs (Domain-specific languages)**

doQuery() functions, 34

doStore() functions, 34

DRY (Don't Repeat Yourself) principle, 276–277

DSLs (Domain-specific languages). See *also* **Flex, 10–11**

E

Eclipse, Flex plug-in for, 388

ECMAScript vs. ActionScript, 365–366

edit template, BlogEntryController, 125–127

edit view, defining, 124–127

element style manipulation (MochiKit), 357

enable_visit_plugin method, 162

encryption, password (Tblog), 145–148

entries list, adding return link to (Tblog), 157–159

error handling

- controller, 223–227
- error_handler decorator, 180–181

event handlers (FBlog), 388–389

exclude method (Django), 79

expose decorator, 181

eXtensible Markup Language (XML).
See **XML (eXtensible Markup Language)**

F

FastData extension (TurboGears), 63

FBlog application. See **application examples (FBlog)**

FeedController class (TurboGears), 234–235

field classes (Django models package), 249–251

filters

- Django template, 257–258
- method (Django), 79

Firebug extension, 230–231

Firefox browser, 230

Flash, 360–361

Flash Player plug-in, 363

Flash-based containers, 364

Flash-based interfaces. See **Flex**

flatpages contrib add-on (Django), 320

Flex (Adobe)

- background, 359–361
- code tester examples. See **application examples (Flex Code Tester)**
- creating basic applications with, 362
- features of, 362
- Flex Builder, 388
- Flex2-based blog example. See **application examples (FBlog)**
- front-end, 384–385
- IDE for Eclipse, 384
- installing Flex SDK, 362
- user interface containers, 364
- user interface control logic, 364–366
- user interface controls, 363

forms

- handling (FBlog), 391

library (Django), 285
 processing (Dblog), 266–269
 widget, defining, 175–177

G

generic views (Django), 284–285

GET method

Django, 79
 HTTP, 24

getComments

function (Ajax blog), 318–320
 path, defining, 197

GMail vs. Mozilla Thunderbird, 8

Google Maps API, 212, 231

Graphic Interchange Format (GIF), 361

Group object (Identity framework), 142

H

handleCallback function, 69

handleChange function, 69

Hansson, David Heinemeier, 93, 206

HBoxes, 364

Hello widget. See application examples (Hello Widget)

History button (Django), 84

Holovaty, Adrian, 71

HTML (HyperText Markup Language)

basics, 21–24
 FBlog HTML file, 392
 HtmlPy template engine, 53
 page generator (Python), 6
 page structure (Pets), 30–32

HTTP (HyperText Transfer Protocol)

defined, 23
 methods, 24
 requests and responses, 6–8
 XMLHttpRequest (XHR), 7, 338–340

I

Identity framework

additional components, 144
 application example. See application examples (Tblog)
 configuration options, 143
 model objects, 140–143
 validation predicates, 152–153

identity.require decorator, 144

imports class (Cable Company), 216–217

index view method (Dblog), 255–259

inheritance, template (Django), 276–282

__init__.py file, 243

installing

Apache HTTPD server, 327–329
 Flex SDK, 362
 TGCrud extension, 109
 user management (Dblog), 285–286

interfaces

Flash-based. See Flex
 user container (Flex), 364
 user control (Flex), 363

internal method calls, 10

Ippolito, Bob, 338, 341

J

Java Hibernate, 211

JavaScript

associative arrays in, 350–351
 function of, 25
 interactive interpreter demo, 343–344, 352
 javascript attribute (widgets), 186, 192
 library, 50
 and MochiKit, 338–339
 object comparisons in, 349–350
 Object Notation (JSON). See JSON (JavaScript Object Notation)

JavaScript (continued)

JavaScript (continued)

writing for comment widget, 193–196

writing in dblog.js, 314–318

job model class (Cable Company), 213–214

JSON (JavaScript Object Notation)

blog object, 318

comment data, 318

Django serialization into, 310

K

Kaplan-Moss, Jacob, 71

Kid templates

application (Cable Company), 228–229

creating (Tblog User Comments), 177–178

defined, 52–53

defining (Tblog), 128–129

for embedding Python in XHTML documents, 64

extending for asynchronous calls, 68

extending to display names, 68–69

fundamentals, 103

Kid templating language, 11

master.kid template (Cable Company), 227–229

visitor statistics view (Tblog), 165–166

L

LatestEntries feed class (FBlog), 323–325

Lesser General Public License (LGPL), 14

licensing, open source, 14

list templates

for BlogAuthorController, 114–116

BlogEntryController, 123–124

list views, defining (Tblog), 113–116, 122–124

list-detail generic views (Dblog), 282–284

listings, code. See code listings

location class (Cable Company), 211–213

Logging module, MochiKit, 352

login, user (Dblog), 288–290

M

Macromedia XML (MXML). See MXML (Macromedia XML)

manage.py command, 73–75

manage.py file, 243–244

manual ORM, 51

mashups, 15

master templates

applying, (Tblog), 135–136

defined, 109

defining (Tblog), 132–133

master.kid template (Cable Company), 227–229

metacharacters for URL pattern matching, 260–261

metaprogramming, 11

methods, HTTP, 24

Miner, Wilson, 71

MIT License, 14

MochiKit

capabilities of, 306, 357

defined, 52, 341

defining XMLHttpRequest with, 68

DOM API, 195

fundamentals, 103

integrating with Django, 306–307

interactive interpreter, 349

licensing of, 14

MochiKit.Async module, 342–349

MochiKit.Base module, 349–351

MochiKit.DOM module, 351–352

MochiKit.Logging module, 352

MochiKit.Logging package, 230

- MochiKit.Signal module, 352–355
 - MochiKit.Visual module, 355–356
 - modules, functions of, 341–342
 - reasons for, 338–339
 - in TurboGears, 14
 - mod_python apache configuration, 328**
 - mod_python extension, 327–328**
 - model classes**
 - defining (Tblog), 105–107
 - defining in Django, 76
 - defining in TurboGears, 57–58
 - supporting (FBlog), 381–382
 - user (Dblog), 286–287
 - writing (Dblog), 246–251
 - model component (MVC) pattern. See MVC (model-view-controller) pattern**
 - model objects**
 - application (Cable Company), 210–211
 - editors, 62–63
 - Identity framework, 140–143
 - updating to User SQLObject (Tblog), 145
 - visitor tracking (Tblog), 160–163
 - ModelDesigner tool (TurboGears), 207–209**
 - model.py module (TurboGears), 205**
 - model-view-controller (MVC) pattern. See MVC (model-view-controller) pattern**
 - Mozilla Thunderbird vs. GMail, 8**
 - MVC (model-view-controller) pattern**
 - architecture in Django, 72–73
 - architecture in TurboGears, 53–54, 101
 - basics, 48–49
 - capabilities of, 51
 - components of, 48–49, 205
 - MXML (Macromedia XML)**
 - basics, 362
 - components of, 363
 - implementation (FBlog), 385–387
 - Myghty template engine, 53**
- N**
- namespace support (MochiKit), 357**
 - number/currency formatting (MochiKit), 357**
- O**
- object comparisons in Python, 349–350**
 - object_detail generic view, 283–284**
 - object_list generic view, 282–284**
 - object-relational mapping (ORM). See ORM (object-relational mapping)**
 - onLoad() function, 33**
 - ORM (object-relational mapping)**
 - automated and manual, 51
 - defined, 24
- P**
- page access, restricting in controller (Tblog), 151–154**
 - paginate decorator, 129**
 - parameters attribute (widgets), 186, 192**
 - parseQueryResult() function, 36**
 - parseSearchResult() function, 36**
 - password encryption (Tblog), 145–148**
 - permissions**
 - object (Identity framework), 142–143
 - user (Dblog), 293–297
 - Pets application. See application examples (Pets)**
 - plug-ins**
 - Flex, for Eclipse, 388
 - visit tracking (Tblog), 162–164
 - priority model class (Cable Company), 213–214**
 - Professional Rich Internet Applications: AJAX and Beyond (Wiley), 339, 349**
 - profile view and template, user (Dblog), 290**

PUT method (HTTP)

PUT method (HTTP), 24

Python language

- \$ in expressions, 115
- { } in expressions, 115
- advantages of, 12–13
- associative arrays, 350–351
- attributes, 115–116
- framework diagrammed, 360
- HTML page generator, 6
- object comparisons in, 349–350
- overview and history, 4–8
- regular expressions, 261

Q

- queryHandler function (Pets), 34–36**
- quickstart controller (TurboGears), 62**

R

RCP (Rich Client Platform) model

- overview of, 4–5
- vs. RIAs, 377–378

Really Simple Syndication (RSS). See RSS (Really Simple Syndication)

record_request method, 162

redirects contrib add-on (Django), 320

registration add-on (Django), 321

regular expressions, Python, 261

render_to_response function, 92

require property (Identity framework), 144

resource model class (Cable Company), 215

resourcelist method (Cable Company), 218–219

RIA (Rich Internet Application) mode).

- See also **Web 2.0 applications**
- application example. See application examples (Cable Company)
- Flex-based. See Flex

- overview of, 7–8
- vs. RCPs, 377–378

Rich Client Platform (RCP) model. See RCP (Rich Client Platform) model

Rich Internet Application (RIA) model. See RIA (Rich Internet Application) model)

Rich Internet Applications: AJAX and Beyond (Wiley), 15

Rich Internet Applications (Wiley), 93

root class (Cable Company), 217–218

root controller (TurboGears)

- basics, 102
- defined, 62
- extended for beView (Tblog), 129–131

RSS (Really Simple Syndication)

- adding to blog application. See application examples (RSS)
- feed capability in TurboGears, 234–237

runserver command (Django), 251

S

save method (Django), 79

save operation, defining, 180–182

saveComment function, 181

script.aculo.us, 53

search view, defining in controller, 69–70

serialization for Ajax, Django, 307–311

service-oriented architecture (SOA), 15

settings.py file, 243, 245

show templates for

BlogAuthorController, 117–118

show views (Tblog), 117–118

Signal module, MochiKit, 352–355

simple.direct_to_template generic view, 284

simple.redirect_to generic view, 284

sitemap contrib add-on (Django), 321

sites contrib add-on (Django), 321

SQL (Structured Query Language)
 basics, 24
 overview of, 24
 sql subcommand (TurboGears), 58

SQLAlchemy, 53, 94

SQLite database file, 244

SQLObject
 attributes, 106–107
 in blog application, 144
 defined, 53–54
 documentation for, 101
 licensing of, 14

startapp command (Django), 245

states, creating (FBlog), 389–390

statistics
 page, adding link to (Tblog), 166–168
 and Visit framework, 157–159
 visitor statistics view (Tblog), 165–166

status model class (Cable Company), 215

Stephenson, Neal, 4

Structured Query Language (SQL). See SQL (Structured Query Language)

style sheets, defining (Tblog), 134–135

syncdb command (Django), 77, 80, 244, 249

syndication contrib add-on (Django), 321

syndication feeds, 321

T

tables
 data view, displaying, 233–234
 defining related (Pets), 39–40

tagging add-on (Django), 321

Tblog application. See application examples (Tblog)

templates
 adding path to controller, 186–188
 application, defining (Django), 87–89
 applying master (Tblog), 135–136

attribute (widgets), 186, 192
 configuring application for (Django), 86–87
 default for data manipulation, 50
 defined in Django, 72–73
 defining for comment widget, 192–193
 defining master (Tblog), 132–133
 edit (Tblog), 125–127
 engine, 50
 incorporating in controller, 178–180
 inheritance (Django), 276–282
 list (Tblog), 114–116, 123–124
 master, defined, 109
 restricting element display in (Tblog), 156–157
 show (Tblog), 117–118
 tags, Django, 256–257
 using widgets in, 186–187
 visitor tracking (Tblog), 164–166

testing, framework support of, 51

TextField widget, 111

tg_flash parameter, 181

tg-admin command (TurboGears)
 creating database with, 58
 creating project with, 54–57, 105
 toolbox command, 104

TGCrud extension (TurboGears)
 code generated by, 108–109, 119
 creating author views with (Tblog), 110–118
 creating entry views with (Tblog), 118–127
 defined, 63
 installing, 109
 overview of, 108

TurboGears framework
 advanced application of, 203–204
 application examples. See application examples
 CatWalk tool, 209–210
 components of, 52–53, 101–104
 vs. Django framework, 11–12

TurboGears framework (continued)

- extensions, 63
- features of, 11–12
- history of, 52
- interview with developer, 93–97
- licensing of, 14
- model component in, 101
- ModelDesigner tool, 207–209
- MVC architecture in, 53–54
- overview of, 13–14
- Toolbox, 104, 206
- views, 103

U

updateCommentsDiv function, 194–195

URL mappings

- in Dblog application, 260–264
- defined in Django, 72–73

urlpatterns

- regular expression processing, 264
- URL pattern matching, 260–261
- variable, 88–89, 260

urls.py file, 243

users

- authenticating (Dblog), 287–290
- comments, accepting, 174–177
- creating (Dblog), 297–302
- identity framework. See Identity framework
- interface containers (Flex), 364
- interface controls (Flex), 363–366
- login and profile (Dblog), 288–290
- management, installing (Dblog), 285–286
- model (Dblog), 286–287
- object (Identity framework), 141–142
- permissions (Dblog), 293–297
- User Comments application, TBlog. See application examples (Tblog User Comments)

utility functions (Pets), 33

V

validate decorator, 181

validation predicates (Identity framework), 152–153

Valverde, Albert, 96

variables

- braces surrounding, 87
- urlpattern, 260

VBoxes, 364

view component (MVC)

- basics, 48–49, 205
- in Cable Company application, 230–234
- capabilities of, 50
- displaying map data, 232–233
- displaying table data, 233–234
- getting data, 232
- map loading, 230–231
- setting up and debugging, 230–231

views

- adding to URL list (Django), 89–90
- calling from controller (TurboGears), 66–67
- composition, 50
- defined in Django, 72–73
- defining tailored, 64–66
- Django application, defining, 90–92
- generic (Django), 284–285
- TurboGears, 103, 205

Visit model object, 160

Visit Tracking framework

- application example. See application examples (Tblog)
- basics, 159–160
- configuration options, 161
- cookies, 138–140
- model objects, basics, 160–161
- plug-ins, 162
- VisitIdentity table, 160–161

Visual module, MochiKit, 355–356

W**Web 2.0 applications**

- capabilities of, 8–10
- defined, 4
- design patterns for, 15–17
- history of, 6–10

web application frameworks

- common capabilities of, 50–52
- MVC in, 48–49

web application path

- modifying for author views (blog application), 110
- modifying for entry views (blog application), 119
- modifying for Kid template (blog application), 129–131

Web publishing (WP), 5**web sites, for downloading**

- CherryPy framework, 53
- Django add-ons, 321
- Django deployment options, 327
- Dojo Toolkit, 53
- Firebug extension, 230
- Flex SDK, 362
- Google Maps API Key, 231
- HtmlPy template engine, 53
- Kid template, 52
- map-handling code, 227
- MochiKit, 52
- mod_python extension, 327
- script.aculo.us, 53
- SQLAlchemy, 53
- SQLObject, 53
- TGCrud, 109
- widgets, 104
- YAML serializer, 311

web sites, for further information

- CherryPy documentation, 102
- decorator function, 102
- Django design philosophy, 71, 241

- Django documentation, 249–250

- Django forms library, 285

- Django generic views, 285

- Django template filters, 257

- Django template tags, 256–257

- JavaScript reference, 24

- MochiKit documentation, 103

- Python regular expressions, 261

- SQLObject documentation, 101

- TurboGears design philosophy, 71

- W3C standard, 21

Web technologies

- application example. See application examples (Pets)

- CGI, 25

- HTML, 21–23

- HTTP, 23–24

- JavaScript, 25

- SQL, 24

- XML, 19–21

widgets (TurboGears)

- attributes, 111

- available types, 121–122

- BlogComment model object, defining, 173–174

- comment widget. See comment widget

- creating Kid template, 177–178

- defining in blog application, 111–113, 119–122, 185–186

- defining template, 192–193

- elements of, 190

- form widget, defining, 175–177

- fundamentals, 104, 169–170

- Hello Widget application. See application examples (Hello Widget)

- user comments, accepting, 174–177

- widget browser, 172

Willison, Simon, 71

XML (eXtensible Markup Language)

X

XML (eXtensible Markup Language)

- basics, 19–21
- blog object, 310
- Django serialization into, 310
- error response (Pets), 37–38
- parsing functions (Pets), 36–37
- response example (Pets), 37

- XMLHttpRequest (XHR), 7, 338–340
- xmlrpc add-on (Django), 321

Y

Yahoo! UI Library (YUI), 341

YAML (YAML Ain't Markup Language)

- blog object, 311
- Django serialization into, 311