

# Index

## A

**abstract Profile Provider methods**, 300

**abstract Roles Provider methods**,  
297–298

**abstraction classes**

in Business Logic Layer, 397–398,  
408–411

database layer and, 377, 393–394

**abstractprovider**, 293–297

**ActiveModules**, 65

**ADA (American Disabilities  
Association)**, 15

**administration**

of hosts. See host administration  
of portals. See portal administration  
Search Admin, 101  
security roles in, 123  
user roles in, 105

**advanced settings**

for hosts, 150–152

for pages, 121

for sites, 115–118

**AffiliateController**, 228–231

**AffiliateInfo class**, 227–231, 235

**affiliates as vendors**, 134

**AJAX (Asynchronous JavaScript and  
XML)**

asynchronous callbacks and, 316, 318

Client API-enabled controls and, 323

ControlMethods and, 325–326

DNN compatibility with, 306–307

**Alexa**, 33

**All Users role**, 105

**Alstad, Kent**, 8

**American Disabilities Association  
(ADA)**, 15

**Andrews, Bryan**, 32

**Announcements module**

defined, 104

Documents module vs., 203

for host administration, 158

Local Resources in, 332

in Project modules, 200

template tokens for, 202

**APIs (application programming  
interfaces)**, 243–285

Client. See Client APIs (application  
programming interfaces)

event logging generally, 243–244

EventLogController in, 244–250

Exception Handling, 253–258

ExceptionLogController in,  
250–253

HTTP modules generally, 260–263

introduction to, 243

localization. See localization APIs

LogException method in, 257–258

of Logging Provider, 244

module interfaces and. See module  
interfaces

Personalization HTTP modules,  
268–269

ProcessModuleLoad

Exception method in, 254–256

ProcessPageLoadException method  
in, 256

ProcessSchedulerException  
method in, 258

Provider Model. See Provider Model  
APIs (application programming  
interfaces)

Scheduler and, 258–260

URL rewriter and, 263–268

Whidbey, 27

**appearance. See also styling of  
modules**

of hosts, 149

of modules, 189, 432

of sites, 115–116, 121

**application programming interfaces  
(APIs). See APIs (application  
programming interfaces)**

**Application Resources**, 331–332

**architecture**, 223–242

Business Logic Layer in, 233–234

Custom Business Object in,  
226–231

Data Access Layer in, 234–236

Data Layer in, 236–237

installation scripts in, 236

introduction to, 223

of namespaces, 240–242

overview of, 231–232

Presentation Layer in,  
232–233

Provider Model in, 224–226

security in, 238

summary of, 240–242

syntax of scripts in, 237

technologies used in, 223–224

upgrade scripts in, 236

**ASCX methods**

DisplaySuggestions, 430–435

EditSuggestions. See

EditSuggestion.ascx

ModuleActions and, 456

Settings, 444

for skinning, 468

**ASP.NET**

and DNN architecture, 223

in evolution of DNN, 2–5

HTTP modules and, 260–261

Membership and, 290–293

pages vs. DNN modules in,  
185

postbacks in, 304–305

**ASP.NET 2.0**

and DNN 4.0, 45–47

in evolution of DNN, 42–43

security in, 238–239

**ASP.NET 2.0 Master Pages**, 460

**ASP.NET 3.5**, 289–290

**asynchronous callbacks**, 316

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

**auditability**, 25

**August Capital**, 68–69, 72

**Augustin, Larry**

as angel investor, 55–56

on Board of Directors, 72–73

fundraising and, 61

Nagiah and, 64

**Australian DotNetNuke (DNN)**

**community**, 41–42

**authentication**

for access to databases,  
78–81

in ASP.NET 3.5, 289

modules for, 101

SystemDotNetNuke for,  
179–180

**authority**, 128–130

**authorization**, 289

**Auto installation method**,  
83–86

## Balmer, Steve

### B

**Balmer, Steve**, 45

**banner advertising**  
modules for, 101, 195–196  
of vendors, 130–133

**Baron, Andy**, 8–9

Base Method Implementations, 444

**Batch Install Extensions**, 200–201

**BAY.NET User Group**, 68–69

**Beadle, Phil**, 44

**Benefactor Program**, 51

**Berkeley Software Distribution (BSD)**  
license, 13, 32–33

**BLL (Business Logic Layer). See Business Logic Layer (BLL)**

**Blog modules**, 104, 200–202

**branding**, 11, 37–39

**Brinkman, Joe**  
creating DNN conferences, 60  
as co-founder of DNN Corporation, 54  
on DNN Board of Directors, 27, 53  
on DNN open source project, 19  
at DNN OpenForce Europe, 62  
on jQuery, 153, 155  
at Tech Ed, 40  
version 3.0 and, 29

**BSD (Berkeley Software Distribution)**  
license, 13, 32–33

**Building Websites with VB.NET and DotNetNuke 3.0**, 35, 41

**Burzi, Francisco**, 11

**business considerations**, 352–354

**Business Logic Layer (BLL)**, 397–418  
business objects for, creating using controller classes, 407–411  
Custom Business Object helper classes in, 405  
developing generally, 397–398  
in DNN architecture, 233–234  
IHydratable interfaces and, 405–407  
introduction to, 397  
IPortable interfaces and, 413–416  
ISearchable interfaces and, 412–413  
optional interfaces for info classes, 405–407  
optional interfaces for  
  SuggestionController class in, 411–412  
properties for info classes in, 398–407  
SuggestionController class in, 408–409  
SuggestionIdController class in, 410–411  
SuggestionIdInfo business object, 402–403

SuggestionInfo business object, 398–401

SuggestionsDisplay  
  Controller class in, 411  
SuggestionsDisplayInfo business object, 403–404  
SuggestionTypeController class in, 409–410  
SuggestionTypeInfo business object, 401–402  
summary of, 416–417

### C

**caching**, 191, 233

**Calculated column**, 209

**Callback. See Client Callback**

CallBackTypeCode Enumerator, 319

**Calvert, Simon**, 43

**Cambrian Explosion**, 62

**Campbell, Richard**, 35

**Caron, Dan**  
adding integrated exception handling and event logging, 25  
on DNN Board of Directors, 27  
on DNN open source project, 19  
enhancements of, 36  
at Professional Developers Conference, 44  
retirement of, 53  
at Tech Ed, 40

**CAS (Code Access Security) environments**, 29

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

**CBO (Custom Business Object). See Custom Business Object (CBO)**

**Charitable Donations business model**, 50

**child pages**, 96

**child portals**, 93–95, 156

**Christmas**, 6

**Circle Graphics**, 22

Class Events region  
  in Edit control, 452–453  
  in Settings module, 445

**Client APIs (application programming interfaces)**, 303–327  
Callback in. See Client Callback  
client-side communications to server, 311–313, 315–316  
control methods of, 323–326  
defined, 305–307  
introduction to, 303  
postbacks and, 304–305  
script caching, client-side, 310–311  
server-side communications to client, 311, 313–315  
summary of, 326–327  
using, 307–310  
view states and, 304–305  
web controls with, 324–325

**Client Callback**, 318–323  
client-side callback handling, 320–321  
introduction to, 316  
life cycle of, 317  
Page\_Init in, 320–321  
registration for, 319–320  
response handling, 322–323  
server-side callback event handling, 322  
setting up, 318–319

**client-side functions**  
callback handling, 320–321  
communications to server, 311–316  
scripts, 232–233  
widget framework, 492–493

**Code Access Security (CAS) environments**, 29

**code-behind regions**  
introduction to, 428–430  
in secondary View control, 434–436  
in Settings module, 443–446

**CodePlex**, 64–65

**CodeSmith**, 374

**columns in Forms & Lists**, 208–210

**commercial evolution of DotNetNuke**, 527–534  
business software, high cost of, 528–529  
introduction to, 527  
philosophy of DNN, 531–532  
present-time advantages of DNN, 529–531  
price/performance improvements in software industry, 527–528  
rationale for commercialization of DNN, 532–534

**concrete data provider**, 386–393

**Concrete Profile Provider**, 300–302

**connection strings**, 90

**Connelly, Cathal**, 44–45

**Connery, Rob**, 62

**containers**  
default, 111–112  
in host administration, 180  
portals and, 98–99  
in Presentation Layer, 232–234  
skinning, 472–475

**Content Panes**, 459

**control code-behind classes**, 441–442

**Control Panel**  
for module layout, 188  
for modules on pages, 187  
for portal administration, 108–109

**controller classes in Logging Provider**, 244

## DotNetNuke (DNN) Corporation

- controls enabled by Client API,** 323–326
  - Controls region of Settings module,** 443
  - convention over configuration,** 503
  - copyright issues,** 32–33
  - core APIs (application programming interfaces).** *See* **APIs (application programming interfaces)**
  - Core Team**
    - enhancement production of, 18–19
    - in evolution of DNN, 13–16, 26
    - introduction to, 13–16
    - reorganization of, 26
    - Trustee roles in, 43–44
    - XXL fork and, 16–17
  - Core Team Trustees,** 44
  - country codes,** 329–330
  - credibility,** 40–41
  - CSS (Cascading Style Sheets)**
    - in localizing images, 348–349
    - in Presentation Layer, 422–423
    - Stylesheet Editor and, 118–119
  - Custom Business Object (CBO)**
    - in architecture, 226–231
    - in Business Logic Layer, 405
    - Hydrator, 229–231, 405–407
    - introduction to, 226–229
    - logging property names and values of, 245–250
  - Custom installation method,** 83
  - custom ModuleAction,** 277–278
  - Custom Widgets,** 497
  - customers first,** 531
  - CVS source-control system,** 15
- D**
- DAL (Data Access Layer),** 234–236, 356
  - Dashboard,** 183–184
  - data abstraction layer,** 393–394
  - Data Access Layer (DAL),** 234–236, 356
  - Data Layer**
    - in architecture, 236–237
    - installation scripts in, 236
    - syntax of scripts in, 237
    - upgrade scripts in, 236
  - data model for users and roles,** 291–292
  - data providers**
    - API, 225–226, 233–234
    - concrete, 386–393
    - in module development, 353
    - multiple, 353
    - resource files with, 331
    - Starter Kit and, 372–373
  - data types,** 208–210
  - database layer,** 377–395
    - concrete data provider in, 386–393
    - data abstraction layer of, 393–394
    - database design generally, 378
    - introduction to, 377–378
    - summary of, 394–395
    - WROX\_AddSuggestion stored procedure, 379–380
    - WROX\_AddSuggestionType stored procedure, 385
    - WROX\_DeleteSuggestion stored procedure, 380–381
    - WROX\_DeleteSuggestionType stored procedure, 385–386
    - WROX\_GetSuggestion stored procedure, 381
    - WROX\_GetSuggestionIDs stored procedure, 381
    - WROX\_GetSuggestionModulesAllTabs stored procedure, 383
    - WROX\_GetSuggestionModulesThisTab stored procedure, 383–384
    - WROX\_GetSuggestions stored procedure, 381–382
    - WROX\_GetSuggestionTypes stored procedure, 386
    - WROX\_Suggestion table, 378–379
    - WROX\_SuggestionType table, 384–385
    - WROX\_UpdateSuggestion stored procedure, 382
    - WROX\_UpdateSuggestionType stored procedure, 386
  - database scripts,** 360
  - databases**
    - logins for, creating, 78–81
    - required for installing DNN 5, 76
    - setting up, 362–363
    - in SQL Server 2005, 76–78
  - debugging,** 366–368
  - delegates,** 269
  - DevConnections,** 60, 62
  - development environment considerations,** 354–355
  - diminishing return,** 15
  - display functions**
    - ASCX, 430–435
    - in Forms & Lists, 215–216
    - for modules, 192–193
  - DisplaySuggestions.ascx**
    - code-behind classes of, 434
    - code-behind regions of, 434–435
    - control regions of, 435
    - resource files of, 433–434
    - user controls of, 430–432
  - distribution,** 505–525
    - creating extensions for. *See* **extensions**
    - extensions model for, 505–506
    - introduction to, 505
    - manifest files. *See* **manifest files**
    - module development for, 374–375
    - Package Creation Wizard, 514–515
    - summary of, 525
  - DLA Piper,** 54, 55
  - DNN (DotNetNuke) 5.0. See DotNetNuke (DNN) 5.0**
  - DNN-Blue skin,** 97
  - dnn.com domain name,** 66–67
  - DNNMembership,** 268
  - doCallBack,** 320–321
  - Document Object Model (DOM),** 312
  - Documents module,** 104, 203–205
  - domain names,** 66–67
  - DotNetNuke (DNN) 2.0,** 22
  - DotNetNuke (DNN) 3.0,** 27–32
  - DotNetNuke (DNN) 3.x,** 56
  - DotNetNuke (DNN) 4.0,** 45–47, 239–240
  - DotNetNuke (DNN) 5.0**
    - administration pages and modules in, 192–193
    - AJAX library and, 325
    - architecture of. *See* **architecture**
    - callback handling in, 319
    - Cambrian Explosion and, 62
    - Cleanup component in, 519
    - Client API in, 306–307
    - CodeSmith and, 374
    - in evolution of DNN, 73
    - extensions in, 505–506
    - HTTP modules in, 263
    - human-friendly URLs and, 266
    - installing. *See* **installing DotNetNuke**
    - introduction of, 73
    - JavaScript in, 310
    - jQuery and, 153, 307
    - LINQ and, 356
    - manifest file format in, 515
    - MembershipProvider functionality in, 293
    - module architecture and directions, 375
    - module interfaces in, 270, 284–285
    - ModuleAction menu in, 275–276
    - Package Creation Wizard in, 514
    - Package Settings section in, 364
    - Profile Provider in, 300
    - Roles Provider in, 298
    - skinning in. *See* **skinning**
    - token replacement engine in, 343
    - Widget Framework in, 307
  - DotNetNuke (DNN) Benefactor Program,** 51
  - DotNetNuke (DNN) brand,** 11–12, 37–38
  - DotNetNuke (DNN) Corporation,** 54–55, 63

## DotNetNuke (DNN) Forge

**DotNetNuke (DNN) Forge**, 65, 222  
**DotNetNuke (DNN) layouts**, 500–502  
**DotNetNuke (DNN) Library**  
 CBO Hydrator in, 229  
 EventLogController in, 244  
 in Host Extensions, 179, 181  
 Library Package Extension, 525  
 PortalModuleBase class in, 425  
 replacing, 371  
**DotNetNuke (DNN) Marketplace**,  
 57–58, 222  
**DotNetNuke (DNN) OpenForce Europe**,  
 62  
**DotNetNuke (DNN) SLA Program**, 63  
**DotNetNuke (DNN) Tree control**,  
 312–316  
**DotNetNuke (DNN) Web Controls**  
 project, 306, 325  
**DotNetNuke (DNN) Web Utility project**,  
 306, 325  
**DotNetNuke (DNN) XXL**, 16  
**DotNetNuke.Common**, 242  
**DotNetNuke.Data**, 242  
**DotNetNuke.Data.SqlDataProvider**, 236  
**DotNetNuke.Entities**, 242  
**DotNetNuke.Framework**, 242  
**DotNetNuke.Modules**, 242  
**DotNetNuke.Schema.SqlDataProvider**,  
 236  
**DotNetNuke.Security**, 242  
**DotNetNuke.Services**, 242  
**DotNetNuke.Setup.SqlDataProvider**,  
 236  
**DotNetNuke.UI**, 242  
**DotNetRocks!**, 35  
**Dreamweaver**, 460, 464  
**Dual License business model**, 49  
**Duffy, Jim**, 35, 45  
**duplicate modules**, 110

## E

**Edit control**  
 control resources, setting with,  
 449–450  
 EditSuggestion.ascx.vb code regions  
 of, 451–454  
 introduction to, 446–449  
 styling of modules with,  
 449  
**EditSuggestion.ascx**  
 code region of, 451–454  
 code-behind class of, 451  
 controls of, 446–449  
 resource files of, 449–450  
**Egan, Dan**, 35  
**Eminem**, 23  
**empty panes**, 469–470  
**encryption**, 295

**End User License Agreement (EULA)**  
 defined, 2  
 of IBuySpy portal, 5  
 licensing vs., 12  
**English language, 172. See also**  
**localization APIs**  
**en-US locale. See localization APIs**  
**EULA (End User License Agreement).**  
**See End User License Agreement**  
**(EULA)**  
**event handlers**  
 registering, 275–276  
 in WROX.DisplaySuggestions View  
 module, 435–436  
 in WROX.Suggestion default View  
 module, 429  
**event logging APIs (application**  
**programming interfaces),**  
 243–253  
 in Business Logic Layer, 233  
 controller classes introduced, 244  
 defined, 244  
 EventLogController, 244–250  
 ExceptionLogController,  
 250–253  
 Logging Provider as, 243–244  
**Event Viewer**, 140–141  
 EventLogController, 244–250  
**eventMessage**, 521–523  
**Events module**  
 advanced features of, 207  
 defined, 104  
 overview of, 204–206  
**evolution of DotNetNuke**  
 architecture in, 223  
 ASP.NET in, 2–5, 42–43  
 awards and accolades for, 61–62  
 benefactor program in, 48–51  
 branding in, 37–39  
 CodePlex, 64–65  
 commercial viability in. *See*  
 commercial evolution of  
 DotNetNuke  
 conferences on, 59–60  
 Core Team in, 13–16, 26, 43–44  
 corporation formation in, 53–55  
 credibility and, 40–41  
 emergence of DNN, 11–12  
 enhancements in, 18–20  
 free module promotion in, 58–59  
 fundraising for, 53, 60–61, 63–64  
 IBuySpy in, 2–9  
 infrastructure in, 36  
 intellectual property issues in, 17,  
 32–33, 41–42  
 introduction to, 1–2  
 IP disputes in, 66–68  
 licensing, 12–13  
 marketing and, 33–35

Marketplace created, 56–58  
 Membership API in, 27  
 Microsoft and, 9–11, 35–36, 44–45  
 MVP (Microsoft Valuable  
 Professionals) program, 60  
 open source philosophy in, 24–25  
 OpenForce, 62–63, 70–71  
 opportunists and, 51–53  
 performance optimization in, 56  
 Professional Edition, 71  
 provider model in, 23–24  
 release schedule, 31  
 security issues in, 20–22, 65–66  
 Series A announcement in, 72–73  
 SLA Program in, 63  
 Slashdot on, 47–48  
 sponsorship during, 17–18  
 stability of applications in, 25  
 subscription fiasco during, 8–9  
 summary of, 73  
 technical education for, 40  
 term sheets in, 68–70  
 third-part components in, 26  
 Web Hosters in, 29  
 DNN web site, 22–23  
 XXL fork in, 16–17  
**evolution of Member Role**, 287–293  
**Exception Handling APIs (application**  
**programming interfaces),**  
 253–258  
 Exceptions class in, 254  
 introduction to, 253  
 LogException method, 257–258  
 ProcessModuleLoadException  
 method, 254–256  
 ProcessPageLoadException  
 method, 256  
 ProcessSchedulerException  
 method, 258  
**exception handling in module**  
**development**, 454–455  
**exception management**, 233, 268  
 ExceptionLogController, 250–253  
**Exceptions class**, 254  
 ExportModule, 413–415  
**exposed methods and properties**,  
 427–428  
**Express tools**, 46  
**extensibility**, 531  
**extensions**, 179–184  
 Authentication System, 512–513  
 configuring, 507–514  
 containers, 180, 507  
 in host administration, 179–184  
 introduction to, 179–180, 506–507  
 language packs, 181, 513–514  
 in Library, 181  
 model of, 505–506  
 modules, 182, 507–511

## IModule Listener

in portal administration, 122  
 providers, 182, 512–513  
 skin objects, 183–184, 511–512  
 skins, 507

**F**

**FAQs**, 104, 205–208  
**Feed Explorer**, 101  
**Feedback module**, 104, 206–208  
**File Manager**  
 FTP with, 138–139  
 introduction to, 137  
 module, 101  
 uploading files and, 138  
**file transfer protocol (FTP)**, 138–139  
**files**  
 managing. *See* File Manager  
 organization of, 461–462  
 permissions, 82  
 type extensions of, 203  
**Fill methods**, 405–407  
**Fleury, Marc**, 53  
**forking**, 16  
**Forms & Lists**, 208–217  
 columns in, 208–209  
 data types in, 208–209  
 display options in, 215–216  
 module-specific settings for, 214  
 portable formats, transferring layouts  
 and data with, 217  
 records, adding via forms, 209–213  
 security permissions in, 214  
**Forum module**, 104  
**framework category**, 38–39  
**Franklin, Carl**, 35  
**Free Software Foundation**, 13  
**friendly URLs**, 264, 266  
**FTP (file transfer protocol)**, 138–139

**G**

**Garrou, Blair**, 53–54  
**Gates, Bill**, 19  
**Germany**, 16, 424  
 GetCallBackEventReference  
 parameters, 318–319  
 GetString method, 333–336  
 GetSystemMessage method, 337–342  
**Global Resources**, 332  
**globalization issues**, 170–172  
**Goldman Sachs report of 2004**, 529  
**Google AdSense**, 101, 197  
**GotDotNet**, 15  
**Guleri, Tim**, 69  
**Guthrie, Scott**, 9–11, 43

**H**

**Haack, Phil**, 62  
 HandleClientCallbackEvent method,  
 322  
**Hanselman, Scott**, 62  
**Healy, Joe**, 70  
**Help module**, 104  
**helper functions**, 454–456  
**Henning, Jon**  
 as Client API project lead, 326  
 at Professional Developers  
 Conference, 44  
 version 3.0 and, 29  
**Highland Capital Partners**, 68–69  
**history of DotNetNuke (DNN). *See***  
**evolution of DotNetNuke**  
**Hopkins, Bruce**, 15, 44  
**Host Accounts**, 93, 94  
**host administration**, 145–184  
 appearance of hosts in, 149  
 containers and, 180  
 Dashboard, 183–184  
 defining hosts, 145–147  
 details of hosts, setting, 147–148  
 expired portals, 157  
 extensions in, 179–184  
 globalization issues of, 170–172  
 Host File Manager, 163  
 Host Module Definitions, 157–163  
 Host SQL page, 164  
 introduction to, 145  
 jQuery for, 153–155  
 language packs in, 169–170, 181  
 languages, 169–172  
 Library in, 181  
 lists for, 173–174  
 modules in, 160–163, 182  
 Package Settings in, 160  
 of payments, 149–150  
 portals and, 155–157  
 providers in, 182  
 Scheduler, 164–169  
 Search Admin for, 172  
 settings for, 147, 150–153  
 skins in, 183–184  
 summary of, 184  
 SuperUser accounts, 174–177  
 vendors in, 164  
**Host Extensions**, 179–184  
**Host File Manager**, 163  
**Host Module Definitions**, 157–163  
**Host Settings**, 101  
**Host SQL page**, 164  
**Howard, Ron**, 8  
**HTML skinning method**  
 ASCX method and, 468  
 client-side widget framework in,  
 492–493  
 for containers, 472–475  
 core DNN classes in, 475–476  
 DNN skinning engine vs., 461  
 empty panes in, 469–470  
 file organization for, 461–462  
 IDs in, 469  
 inheritance in stylesheets and,  
 475–476  
 introduction to, 459–460  
 for layouts, 468–469, 470–472  
 Master Pages vs., 460  
 parsing in, 462–464  
 rounded corners in, 502  
 skin preview images, creating, 503  
 skinobjects, adding functionality with,  
 476–477  
 styles guide for, 475–476  
 YUI Grids, Fonts, and Reset libraries  
 in, 498–500  
**HTTP modules**, 260–269  
 in DNN generally, 263  
 DNNMembership, 268  
 Exception Management, 268  
 introduction to, 260–263  
 Personalization, 268–269  
 URL rewriter, 263–268  
 Users Online, 268  
**HTTP Pipeline**, 260–261, 264  
**Hummer Winblad**, 64  
**hydrators**  
 Custom Business Object, 405–407  
 introduction to, 229–230  
 using, 230–231

**I**

**IActionable**, 270–271  
**IActionControl**, 284–285  
**IBuySpy**  
 Portal, 2–5  
 Portal Forum, 5–6  
 Workshop, 5–9  
**ICANN (Internet Corporation for  
 Assigned Names and Numbers)**,  
 67  
 IClientAPICallbackEventHandler,  
 318–320  
**IDE (Integrated Development  
 Environment)**, 46  
**IDs in skinning**, 469  
**IFrame**, 104, 217–218  
**IHydratable interface**, 405–407  
**IIS (Internet Information Server)**,  
 82–83, 355–356  
**Image module replacement**, 217  
**images, localization of**, 348–349  
**IMC (Inter-Module Communication)  
 framework**, 281  
**IModule Listener**, 281

## IModuleCommunicator

- IModuleCommunicator**, 281
  - IModuleControl**, 284–285
  - `ImportModule`, 415–416
  - independent software vendors (ISVs)**, 351
  - IndexingProvider**, 282–284
  - INETA (International .NET Association)**, 40
  - Info-Tech Decision Diamond award**, 61
  - inheritance**, 269, 475–476
  - Inner Team**, 26
  - Installation Manifest**, 361–362
  - installation scripts**, 236
  - installing DotNetNuke**, 75–91
    - configuring IIS for, 82–83
    - database logins for, 78–81
    - databases in SQL Server for, 76–78
    - downloading software for, 76
    - file permissions in, 82
    - Install Package for, 75
    - options for, 83–86
    - required for, 75–76
    - Starter Kit for, 86–89
    - steps for, 76
    - summary of, 91
    - troubleshooting, 90
    - unzipping packages for, 76
    - upgrading for, 89–90
  - `Instance()` **method**, 235
  - Integrated Development Environment (IDE)**, 46
  - intellectual property**, 17, 32–33
  - interfaces**
    - application programming. *See* APIs (application programming interfaces)
    - in Business Logic Layer, 405–407, 411–412
    - as calling mechanisms, 269–270
    - `IHydratable`, 405–407
    - `IPortable`, 278–280, 413–416
    - `ISearchable`, 281–284, 412–413
    - `IUpgradable`, 280, 521–523
    - of modules. *See* module interfaces
    - in Presentation Layer, 426–428
  - Inter-Module Communication (IMC) framework**, 281
  - International .NET Association (INETA)**, 40
  - Internet Corporation for Assigned Names and Numbers (ICANN)**, 67
  - Internet Information Server (IIS)**, 82–83, 355–356
  - investors**, 61, 64, 68–69
  - `IPortable` **interface**, 278–280, 413–416
  - Iran**, 65–66
  - Isakson, Cory**, 20
  - ISCN (Iran Security Center Networks)**, 66
  - `ISearchable` **interface**, 281–284, 412–413
  - ISkinControl**, 284–285
  - `IUpgradable` **interface**, 280, 521–523
- ## J
- JavaScript**
    - Client API and, 310–311
    - Client API-enabled controls and, 323
    - `ControlMethods` and, 325–326
  - JBOSS**, 529
  - jQuery**
    - Client API and, 307, 311
    - in host administration, 153–155
- ## K
- Kalyani, Nik**
    - creating DNN conferences, 60
    - at BAY.NET User Group, 68–69
    - on Benefactor Program, 51
    - Board membership of, 53
    - on branding of DNN, 37–39
    - on Cambrian name, 62
    - as CEO, 63–64
    - as co-founder of DNN Corporation, 54
    - on DNN 4.0, 46
    - Nagiah and, 68
    - at Professional Developers Conference, 44
  - Kermani, Morteza**, 65–66
  - KEYID property**, 405–407
  - Khan, Omar**, 43
  - Knight Ridder**, 66–67
- ## L
- Language Integrated Query (LINQ)**, 354, 356
  - language translation, 169–172. *See also* localization APIs**
  - law of diminishing return**, 15
  - layouts**
    - CSS and divs creating, 470–472
    - of modules, 188
    - panes and stylesheets controlling, 468–469
  - legacy containers**, 180
  - libraries**
    - AJAX, 325, 493
    - DNN. *See* DotNetNuke (DNN) Library
    - jQuery, 153–155
    - Microsoft Enterprise, 387
    - Web Control, 325
    - Yahoo! YUI, 498–500
  - licensing**
    - Berkeley Software Distribution license, 32–33
    - end users agreements for. *See* End User License Agreement (EULA)
    - in evolution of DNN, 12–13
    - of intellectual property, 32–33
    - Service Provider Licensing Agreement, 35
    - third-party issues in, 26
    - Trademark License Agreement, 42
  - Links module**, 101, 197–199
  - LINQ (Language Integrated Query)**, 354, 356
  - Lists module, 101, 173–174. *See also* Forms & Lists**
  - LiveID**, 62
  - Local Resources**, 332
  - `Local Strings` **region**, 443, 451–452
  - localization**, 329–349
    - APIs for. *See* localization APIs
    - Application Resources in, 331–332
    - in Business Logic Layer, 233
    - Global Resources in, 332
    - of images, 348–349
    - introduction to, 30, 329
    - Local Resources in, 332
    - locales in DNN for, 329–331
    - of modules generally, 343–345
    - in Presentation Layer, 423–424
    - resource files in, 331–332
    - of static strings in ASCX files, 345
    - of static text, 345–348
    - summary of, 349
  - localization APIs**, 332–343
    - `GetString` method, 333–336
    - `GetSystemMessage` method, 337–342
    - introduction to, 332–333
    - token replacement engine, 343–344
  - log classification**, 244
  - log type**, 244
  - Log Viewer**, 101
  - `LogException` **method**, 257–258
  - Logging Provider**, 243–253
    - controller classes in, 244
    - as event logging API, 243–244
    - `EventLogController` in, 244–250
    - `ExceptionLogController` in, 250–253
    - introduction to, 243–244
  - logins**
    - in ASP.NET 3.5, 289
    - controls of, 238
    - database, 78–81
  - logos**, 37–38
  - The Long Tail**, 52
  - Look-up column**, 209

**Loomans, Jeff**, 69  
**Lucarino, John**, 4

## M

**magazines**, 41  
**magical software**, 19  
**Manage Profile pages**, 128–129  
**Management Studio**, 77–78  
**Manifest document**, 14–15  
**manifest files**, 515–525  
   assembly components in, 518  
   Authentication System in, 524–525  
   cleanup components in, 519  
   components of packages generally, 517  
   Config components in, 519–521  
   container components in, 523–524  
   elements and attributes of packages, 515–517  
   file components, 517  
   introduction to, 515  
   language pack components in, 524  
   Library Packages, 525  
   Module components in, 521–523  
   Provider packages, 525  
   script components in, 518  
   skin components in, 523–524  
   Skinobject components in, 524  
**marketing**, 33–35, 114–115  
**MarketShare**, 101  
**Marquardt, David**, 68  
**Masanas, Vicenc**, 29  
**MaximumASP**, 36  
**McCulloch, Scott**, 19, 29  
**Media module**, 104, 217, 219  
**Mediator design pattern**, 275–278  
**Medium Trust Code Access Security (CAS) environments**, 29  
**Mehra, Vivek**, 68–69, 72  
**Member Role**, 287–302  
   applications and, 290–291  
   data model for users and roles in, 291–292  
   introduction to, 287–289  
   Membership Provider and, 292–297  
   overview of, 290–293  
   portals and, 290–291  
   Profile Provider and, 292–293, 299–302  
   Roles Provider and, 292–293, 297–299  
   security in ASP.NET 3.5 and, 289–290  
   summary of, 302  
**MemberRole.dll**, 287–289  
**Membership Subscription business model**, 50–51

**Membership/Roles Provider**  
   applications and, 290–291  
   data model for users and roles in, 291–292  
   Member Role in, 293–297  
   portals and, 290–291  
   security in, 238–240, 289  
   services of, 290  
**menus, displaying**, 276–277  
**meritocracy**, 14  
**Microsoft**  
   CodePlex and, 64–65  
   evolution of DNN and, 9–11, 29  
   Most Valuable Professional summit, 44–45  
   Professional Developers Conference, 20  
   sponsorship agreement with, 17–18  
   SQL Server. *See* SQL Server  
   Tech Ed conference of, 40  
   Windows Shared Hosting Accelerator Program, 56  
**Microsoft ASP.NET**  
   in DNN infrastructure, 1, 36  
   Membership API and, 27  
   provider model in, 23–24  
   sponsorship agreement with, 17–18  
   subscriptions fiasco and, 9–11  
   version 2.0, 42  
**Microsoft Hosting program**, 35–36  
**Microsoft Membership APIs (application programming interfaces)**, 27  
**Microsoft Virtual PC (VPC)**, 355  
**Microsoft.NET Runtime**, 76  
**Minh, Tam Tram**, 32, 46  
**MinimalEntropy skin**, 98  
**MinimalExtropy skin**, 97  
**mission of DotNetNuke (DNN)**, 532  
**Mitchell, John**, 44  
**Model View Controller (MVC) toolkit**, 375  
**modularity**, 531  
**module control definition attributes**, 512  
**module development, 351–376. *See also* modules**  
   beginning steps for, 357  
   business considerations in, 352–354  
   Business Logic Layer in. *See* Business Logic Layer (BLL)  
   configuration in, generally, 359  
   creating from scratch, 358  
   database layer in. *See* database layer  
   distribution and, 374–375  
   in DNN architecture, 375  
   environment for, 354–355  
   existing standard modules, using, 357–358

  interfaces in. *See* module interfaces  
   introduction to, 351–352  
   module setup, 359–362  
   platform choices for, 355–356  
   Presentation Layer in. *See* Presentation Layer  
   project assessment, 352–357  
   public vs. private modules, 354  
   publishing, 357  
   registering modules within portals, 363–365  
   scope of modules in, 354  
   source control systems for, 357  
   Starter Kit for. *See* Starter Kit  
   summary of, 376  
   templates for, 358  
   virtual environments for, 355  
   Visual Studio 2008 in, 365–368  
   web project types, 356–357  
   WROX.Suggestion example of. *See* WROX.Suggestion module  
**module interfaces. *See also* modules**  
   in DNN 5.0, 284–285  
   IActionable, 270–271  
   IModuleListener, 281  
   IModuleCommunicator, 281  
   introduction to, 269–270  
   IPortable, 278–280  
   ISearchable, 281–284  
   IUpgradable, 280  
   ModuleAction API and. *See* ModuleAction APIs  
   summary of, 285  
**ModuleAction APIs**  
   classes in, 272  
   introduction to, 271–275  
   Mediator design pattern and, 275–278  
   module interfaces. *See* ModuleAction APIs  
   properties of, 273  
**modules, 185–222**  
   adding on pages, 187  
   Announcements. *See* Announcements module  
   Banners, 195–196  
   basic settings for, 189–190  
   Blog, 200–202  
   commercial vs. open source  
     third-party, 221–222  
   developing. *See* module development  
   displaying content, generally, 192–193  
   Documents, 104, 203–205  
   duplicate, 110  
   Events, 104, 204–207  
   FAQs, 104, 205–208  
   Feedback, 206–208  
   Forms & Lists. *See* Forms & Lists  
   Google AdSense, 197  
   host administration, 182

## modules (continued)

### modules (continued)

HTTP. See HTTP modules

IFrame, 217–218

installing, 160–163

interfaces for. See module interfaces

introduction to, 185–186

layout of, 188

Links, 197–199

localization of. See localization APIs

managing on pages, 187

Media, 217, 219

module-specific settings for, 191–192

News Feeds (RSS), 217, 219

Package Settings for, 160

Page Settings for, 191–192

portals and, 99–105

Project, 200

recycling, 139–140

Search Input, 199

Search Results, 199–200

settings for, 189–192

sub-project, 220–221

summary of, 221–222

Survey, 217, 219

Text/HTML, 193–195

user content of, 193

user controls in, 232–233

User Defined Table, 208

Users Online, 217–219

XML/XSL for, 220–221

**ModuleSettingBase class**, 442

**Morgenweck, Will**, 65, 70

**Most Valuable Professional (MVP)**, 44–45

**MVC (Model View Controller) toolkit**, 375

**MVP (Most Valuable Professional)**, 44–45

## N

**Nagiah, Navin**

as CEO, 68

as CEO and Board member, 72

hiring of, 64

**names, adding**, 370–371

**namespaces**

of architecture, 240–242

in Business Logic Layer, 402–404

in Controller classes, 409–411

**Nandi, Shawn**, 29

**navigation**, 121–122, 455–456

**.NET assemblies**, 360

**.NET Developers Journal (.NETDJ)**, 47–48

**.NETDJ (.NET Developers Journal)**, 47–48

**New Extension Wizard**, 506–507, 511

**News Feeds (RSS)**, 104, 217–219

**newsletters**, 101, 135–137

**Nguyen, Oliver**, 68–69

**“No Respect for Windows Open Source”**, 47

**node action types**, 520–521

**“nuke” slogan**, 11, 37–38

**Nurse, Charles**

on DNN 3.0, 30

on DNN 4.0, 45

on DNN 5.0, 73

at DNN OpenForce Europe, 62

enhancements of, 36

on Extensions, 122

optimizing for shared hosting environment, 56

at Professional Developers Conference, 44

## O

**Object-Relational Mapping (ORM) tools**, 356

**Onset Ventures**, 68–69

**open source philosophy**

Augustin investing in, 55

balancing with commercial ecosystem, 53

bottom-up movement of, 529–531

business models and, 49–50

commercial ventures vs., 4–5

credibility and, 40–41

in evolution of DNN, 24–25

forking and, 16–17

licensing and, 12–13

media coverage of, 47–48

Microsoft vs., 18

security issues in, 20–21

subscriptions vs., 8–11

**OpenForce North America**, 62–63, 70

**OpenID authentication system**, 62

**ORM (Object-Relational Mapping) tools**, 356

**Outer Team**, 26

## P

**Package Creation Wizard**, 514–515

**package settings attributes**, 507–508

**packages, building**, 515–525

assembly components in, 518

Authentication System in, 524–525

cleanup components in, 519

components of, generally, 517

Config components in, 519–521

container components in, 523–524

elements and attributes of, 515–517

file components in, 517

introduction to, 515

language pack components in, 524

Library Packages, 525

Module components in, 521–523

Provider packages, 525

script components in, 518

skin components in, 523–524

Page\_Init, 320–321

**pages**

management of, 116–117

Page Settings, 191

in portal administration, 95–97, 119–121

**Pane Collapse Widget**, 494–495

**panes**

in modules, 185–186

in portals, 97

in Presentation Layer, 232–234

**parent/child portals**, 93–95

**parsing skins**

ASCX method vs., 468

introduction to, 459, 462–464

ParseSkin Package link for, 475

**passwords**

aging of, 176

in Membership/Roles Provider, 295

for SuperUser accounts, 146

**payments**

host administration of, 149–150

portal administration of, 125–126

Site Settings for, 117–118

**PayPal**, 126

**PDC (Professional Developers Conference)**, 20, 44

**permissions**

in Forms & Lists, 214

setting file, 82

for viewing and editing modules, 189–190

**Personalization HTTP modules**, 268–269

**philosophy of DotNetNuke (DNN)**, 532

**PHP-Nuke**, 11

**platform choices**, 355–356

**PNG Transparency Widget**, 495

**Pointer, Leigh**, 35, 59

**portable formats**, 217

**portal administration, 107–144. See also portals**

authority in, 128–130

configuring portals for, 113–119

control panel for, 108–109

developments in, 142–143

Event Viewer for, 140–141

extensions in, 122

File Manager and, 137–139

introduction to, 107

membership services in, 128

modules in, 188, 192–193

navigational structures in, 121–122

## server-side communications to client

- newsletters and, 135–137
- pages in, 119–121
- Portal Administrator for, 107
- privileges in, 128–130
- public roles in, 128
- Recycle Bin and, 139–140
- security roles in, 123–127
- Site Log in, 134–135
- Site Settings for, 113–118
- Site Wizard for, 109–113, 141
- Solutions Explorer for, 142
- starting, 107–108
- Stylesheet Editor in, 118–119
- summary of, 143–144
- Text/HTML module for, 193–194
- vendors in, 130–134
- Portal Aliases module**, 4, 101
- Portal module**, 101
- PortalID**, 238
- `PortalModuleBase` class, 425–427
- portals**, **93–106**. *See also* **portal administration**
  - cleaning up, 140
  - containers in, 98–99
  - in host administration, 155–157
  - introduction to, 93
  - Member Role and, 290–291
  - modules in, 99–105
  - organization elements, 93
  - pages in, 95–97
  - panes in, 97
  - parent/child, 93–95
  - security of, 239
  - skins in, 97–98
  - summary of, 105
  - user roles for, 103–105
- postbacks**, 304–305, 316
- Presentation Layer**, 419–457
  - code-behind regions in, 428–430, 443–446
  - code-behind regions in secondary View control, 434–436
  - control code-behind class in, 441–442
  - control resources in, 439–441, 449–450
  - CSS styling in user controls in, 422–423
  - in DNN architecture, 232–233
  - Edit control generally, 446–449
  - `EditSuggestion.ascx` in. *See* `EditSuggestion.ascx`
  - exception handling in, 454–455
  - helper functions for, 454
  - introduction to, 419
  - localization in, 423–424
  - navigation URLs, 455–456
  - optional interfaces, 426–428
  - `PortalModuleBase` class, 425–426
  - Settings control, 436–438
  - styling of modules in, 432–433, 438–439, 449
  - summary of, 456–457
  - user controls generally, 419–420
  - View controls in, 421–422, 424–425, 430–434
  - Private Members, 398–404, 429
  - Private Methods, 445–446, 453–454
  - private vs. public modules**, 354, 357
  - privileges**, 128–130
  - `ProcessModuleLoadException`, 254–256
  - `ProcessPageLoadException`, 256
  - `ProcessSchedulerException`, 258
  - product development guidelines**, 14–15
  - Professional Developers Conference (PDC)**, 20, 44
  - Professional DotNetNuke ASP.NET Portals**, 37, 40–41
  - Professional Services business model**, 49–50
  - Profile Provider**, 292–293, 299–302
  - Project modules**, 200
  - Provider Model APIs (application programming interfaces)**
    - in architecture, 224–226
    - configuring, 226
    - in evolution of DNN, 23–24
    - introduction to, 224
    - usage of, 224–225
  - providers**, 182
  - public methods in Controller classes**, 408–411
  - public properties in Business Logic Layer**, 399–404
  - public roles and membership services**, 128
  - public vs. private modules**, 354, 357, 374–375
  - publishing process**, 357

## R

  - Radcliffe, Mark**, 54, 55
  - records, adding via forms**, 209–213
  - Recycle Bin**, 101, 139–140
  - Registered Users**, 105, 123
  - registering modules in portals**, 363–365
  - registration in Client Callback**, 319–320
  - release schedules**, 31
  - Relocation Widget**, 495
  - Reports module**, 104
  - Repository module**, 104
  - reseller environment**, 56–58
  - Resource File Verifier**, 172
  - Resource Files (RESX)**
    - for displaying modules, 433
    - in localization, 329, 331–332, 423–425
    - for setting controls, 439–441, 449
  - response handling**, 322–323
  - RESX (Resource Files)**. *See* **Resource Files (RESX)**
  - Roles Provider**, 292–293, 297–299
  - Rotator Widget**, 494
  - rounded corners**, 502–503
  - RSS module**, 217

## S

  - Santry, Patrick**
    - on Core Team, 26
    - on DNN Board of Directors, 27
    - marketing efforts of, 34
  - Scarbeau, Brian**, 70
  - Scheduler**, 164–169
    - as API, 258–260
    - configuration settings of, 168
    - in host administration, 164–169
    - introduction to, 164–165
    - item details in, 165–166
    - limitations of, 169
    - module, 101
    - status of schedules in, 166–168
  - script caching**, 310–311
  - scripts, database**, 360
  - SDN (Software Developer Network)**, 59–60, 62, 70
  - Search Admin**, 101, 172
  - search engine optimization**, 199, 264
  - Search Input module**, 102, 199
  - Search Results module**, 102, 199–200
  - secondary View control**
    - code-behind class in, 434
    - code-behind regions in, 434–436
    - introduction to, 433–434
    - in Presentation Layer, 430–432
  - security**
    - in ASP.NET, 238–239, 289–290
    - of databases, 78–81
    - in DNN 4.0, 239–240
    - in DNN 5 architecture, 238–240
    - in evolution of DNN, 20–22
    - Membership Provider and, 239–240
    - module for, 102
    - permissions, 82, 214
    - of portals, 239
    - roles, 103, 123–127
    - Site Settings for, 115–116
  - Sellers, Mitchell**, 66
  - server-side callback event handling**, 322
  - server-side communications to client**, 311, 313–315

## Service Provider Licensing Agreement (SPLA)

### Service Provider Licensing Agreement (SPLA), 35

#### settings

- for Client Callback, 318–319
- for control resources, 439–441, 449–450
- for host administration, 101, 150–153
- for modules, 189–192, 436–438
- in secondary View control. *See* Settings control in secondary View control
- for sites. *See* Site Settings

#### Settings control in secondary View control

- code-behind regions in, 443–446
- control code-behind class in, 441–442
- control resources, setting, 439–441
- secondary View control resources, 436–438
- styling of modules, 438–439

SharedResources.resx, 332

Sierra Ventures, 69, 72

Site Log module, 102, 134–135

Site Settings, 113–118

- for appearance of sites, 115–116
- for details of sites, 114
- for marketing of sites, 114–115
- for page management, 116–117
- for payments, 117–118
- for security, 115–116
- for usability, 118

site virtualization, 4

Site Wizard

- creating web sites with, 109–113
- introduction to, 102
- in portal administration, 141

skinning, 459–504. *See also* skins

- advantages of DNN skinning engine, 461
- ASCX method for, 468
- ASP.NET 2.0 Master Pages vs., 460
- client-side widget framework in, 492–493
- containers, creating basic, 472–475
- core DNN classes in, 475–476
- Custom Widgets for, 497
- deploying skins, 503–504
- DNN layouts in, 500–502
- empty panes, handling, 469–470
- file organization for, 461–462
- IDs and classes in, 469
- inheritance in stylesheets, 475–476
- introduction to, 459–460
- layouts in, 468–469, 470–472
- Pane Collapse Widget for, 494–495
- parsing skins, 462–464
- PNG Transparency Widget for, 495
- previewing skins, 503
- production phase of, 462

- references for skinobjects, 477–492
- Relocation Widget for, 495
- Rotator Widget, 494
- rounded corners in, 502–503
- skinobjects for. *See* Skinobjects
- Style Scrubber Widget for, 495–496
- styles guide, 475–476
- Stylesheet Widget for, 496
- summary of, 504
- super stylesheets generally, 498
- Visibility Widget for, 497

#### Skinobjects

- adding functionality with, 459, 476–477
- ASCX method and, 468
- in containers, 472–473
- managing skins with, 466–467
- reference guide for, 477–492

#### skins. *See also* skinning

- creating new, 178
- in host administration, 183
- installing new, 178–179
- in portals, 97–98
- in Presentation Layer, 232–234
- selecting, 111
- skin objects, 183–184

Software Developer Network (SDN), 59–60, 62, 70

software development languages, 353

#### software industry

- advantages of DNN in, 529–531
- costs of business software and, 528–529
- price/performance improvements in, 527–528
- rationale for commercialization of DNN in, 532–534

Solutions Explorer, 142

source control systems, 357

Source Package, 75

SourceForge, 33–34

SPLA (Service Provider Licensing Agreement), 35

Sponsorship business model, 17–18, 49–50

SQL module, 102

#### SQL Server

- as backend database, 76, 291, 377–378
- in Data Layer, 236–239
- and DNN architecture, 224–225
- installing DNN 5 and, 76–78
- in module development, 354–356, 377–378

SQL Server Management Studio, 77–78

SqlDataProvider, 236–237, 372–373

Stallman, Richard, 13

standard modules, 357

Starter Kit

- completing development with, 373–374
- creating projects with, 368–370
- data provider project in, 372–373
- for installing DNN 5, 86–89
- introduction to, 358, 368
- for module development, 358
- names, adding, 370–371
- properties, setting, 371–372

static text, 345–348

Store module, 104

strategic partnerships, 35–36

#### strings

- invalid connection, 90
- localizing, 332, 344
- personalizing and localizing, 337–342
- static, 333–336, 344–348

style classes, 477

Style Scrubber Widget, 495–496

styles guide, 475–476

Stylesheet Widget, 496

#### stylesheets

- controlling layouts with, 468–469
- editor for, 118–119
- hierarchy in DNN skins, 476
- inheritance in, 475–476
- super. *See* super stylesheets
- widgets, 496

#### styling of modules

- with Edit control, 449
- generally, 432–433
- in secondary View control, 438–439

sub-project modules, 220–221

Subscribers security role, 123

Subscribers user role, 105

subscription fiasco, 8–9

Suggestion module. *See*

WROX.Suggestion module

Suggestion.ascx.vb

- edit controls of, 446–449, 451–454
- ModuleActions property in, 456
- optional interfaces of, 426
- resource files of, 424–425
- user controls of, 421–422

SuggestionController class, 408–409

SuggestionIdController class, 410–411

SuggestionIdInfo business object, 402–403

SuggestionInfo business object, 398–401

SuggestionsDisplayController class, 411

SuggestionsDisplayInfo business object, 403–404

SuggestionTypeController class, 409–410

`SuggestionTypeInfo` **business object**, 401–402

**Sugishita, Tomotoshi**, 66

**super stylesheets**

DNN layouts in, 500–502  
introduction to, 498  
rounded corners in, 502–503  
YUI Grids, Fonts, and Reset libraries as, 498–500

**SuperUser accounts**, 145–146, 174–177

**Survey module**, 104, 217–219

**syntax of scripts**, 237

**SYS-CON Enterprise Open Source conference**, 53

## T

**tables**

DNN attributes, supporting, 240  
User Defined, 105, 208  
WROX\_Suggestion, 378–379  
WROX\_SuggestionType, 384–385

**Tabs module**, 95, 102

**TDD (Test Driven Development) approach**, 375

**Tech Ed**, 40

**templates**

in Announcements module, 202  
for modules, 358  
for portals, 156–157  
for skins. *See* templates for skins  
for web sites, 109

**templates for skins**

containers in, 472  
introduction to, 459  
layouts in, 500  
Master Pages for, 460  
rounded corners in, 502  
Skinobjects in, 477

**Test Driven Development (TDD) approach**, 375

**Text/HTML module**, 102, 191–195

**time zones**, 170

**Timer Method mode**, 169

**The Tipping Point**, 52

**token fields**, 211–213

**token replacement engine**, 343–344

**trademarks**

domain names and, 67  
in evolution of DNN, 17  
policies regarding, 41–42  
**trailing periods for file names**, 66  
**training of developers**, 352–353  
**translation of languages. See localization APIs**

**transparency**, 532

**Transparency Widget**, 495

**Tulsa TechFest**, 59

**Typical installation method**, 83

## U

**UDRP (Uniform Domain-Name Dispute-Resolution Policy)**, 67

**Unauthenticated Users role**, 105

**Uniform Domain-Name**

**Dispute-Resolution Policy (UDRP)**, 67

**United States Patent and Trademark Office (USPTO)**, 67

**United States Rehabilitation Act**, 15

**University of California**, 13

**University of Texas**, 15

**unzipping packages**, 76

**Upgrade Package**, 75

`UpgradeModule`, 280

**upgrading scripts**, 236

**upgrading to DotNetNuke (DNN)5**, 89–90

**uploading files**, 138

**URL rewriter**, 263–268

**user accounts**, 77–78, 174–177

**user content**, 193

**user controls**, 238

in ASP.NET 3.5, 289  
CSS styling within, 422–423  
introduction to, 419–420  
localization in, 423–424  
View controls, 421–422  
in WROX.Suggestion module, 360, 436–438

**User Defined Table**, 105, 208

**user management**, 238, 289

**user roles**, 103–105

**UserID**, 291

**Users Online module**

as HTTP module, 268  
introduction to, 105  
statistics on, 217–219

**USPTO (United States Patent and Trademark Office)**, 67

## V

**valid layouts**, 470–472. *See also* layouts

**validation**, 295

**VB (Visual Basic)**, 353, 357–358

**VC (Venture Capitalists). See investors**

**Veenstra, Geert**, 19, 35

**vendors**

as affiliates, 134  
host administration for, 164  
portal administration for, 130–134

**Vendors module**, 102

**Vertical Application business model**, 50

**View control resources**

code-behind class of, 425  
code-behind regions of, 428–430  
introduction to, 424–425  
optional interfaces of, 426–428  
`PortalModuleBase` class of, 425–426  
secondary, 430–432  
styling of modules in, 432–433

**View controls**, 421–422

**view states**, 304–305

**virtual development environments**, 355

**Virtual PC (VPC)**, 355

**Virtual Private Server (VPS) accounts**, 36

**Viscott, David**, 107

**Visibility Widget**, 497

**vision of DotNetNuke (DNN)**, 532

**Visual Basic (VB)**, 353, 357–358

**Visual Studio**

Client API in, 326  
custom module development with, 351–352

database layer and, 378

Starter Kit with, 86, 91, 368–374

**Visual Studio Magazine**, 61

**Visual Web Developer**, 86–89, 91

**Vogt, Anson**, 22–23

**Volunteer business model**, 49

**VPC (Virtual PC)**, 355

**VPS (Virtual Private Server) accounts**, 36

**VSS (Visual SourceSafe)**, 15

## W

**Walker, Bill**, 46, 70

**Walker, David**, 59

**Walker, Shaun**

at Bay.Net User Group, 68  
on Board of Directors, 53, 72  
at DNN OpenForce Europe, 62  
on evolution of DNN, 1  
in IP dispute, 66

**Wanagel, Jonathan**, 65

**WAP (Web Application Project)**, 356–358

**Washington, Michael**, 356

**Web Application Project (WAP)**, 356–358

**web forms**, 232–233

**Web Hosters**, 29

**web portals. See portals**

**web project types**, 356–357

**Web Server**, 76

**web site for DotNetNuke (DNN)**, 22–23

**Web Site Project (WSP)**, 356–358

**Webb, Michael**, 70

## WF (Windows Workflow) services

**WF (Windows Workflow) services**, 354

**Whidbey APIs (application programming interfaces)**, 27

**White, Jeremy**, 29

### widgets

Custom, 497  
 framework of, 492–493  
 Pane Collapse, 494–495  
 PNG Transparency, 495  
 Relocation, 495  
 Rotator, 494  
 Style Scrubber, 495–496  
 Stylesheet, 496  
 Visibility, 497

**Wiki module**, 105

**Wikipedia**, 93

**Willhite, Scott**

on abundance mentality, 50  
 on Board of Directors, 27  
 business acumen of, 52–53  
 as co-founder of DNN Corporation, 54  
 human resources work of, 43–44  
 infrastructure management of, 36  
 at Professional Developers Conference, 44  
 at Tech Ed, 40

**Windows Authentication**, 78–81

**Windows Shared Hosting Accelerator Program**, 56

**Windows Workflow (WF) services**, 354

**The World Is Flat**, 52

**WROX.Suggestion module**

Business Logic Layer in. See WROX.Suggestion module, Business Logic Layer  
 configuring generally, 359, 365–368  
 database layer in. See WROX.Suggestion module, database layer  
 definitions module controls, 512  
 overview of, 358–359  
 Presentation Layer in. See WROX.Suggestion module, Presentation Layer  
 registering, 363–365  
 Starter Kit for. See Starter Kit

**WROX.Suggestion module, Business Logic Layer**

business objects in, creating, 407–411

Custom Business Object helper class in, 405

IHydratable interface in, 405–407

info class interfaces in, 405–407

introduction to, 397–398

IPortable interface in, 413–416

ISearchable interface in, 412–413

properties for info class in, 398–407

SuggestionController class in,

408–409

SuggestionController class

interfaces in, 411–412

SuggestionIdController class in,

410–411

SuggestionIdInfo business object

in, 402–403

SuggestionInfo business object in,

398–401

SuggestionsDisplayController

class, 411

SuggestionsDisplayInfo business

object in, 403–404

SuggestionTypeController class

in, 409–410

SuggestionTypeInfo business

object in, 401–402

**WROX.Suggestion module, database layer**

introduction to, 377–378

WROX\_AddSuggestion stored

procedure, 379–380

WROX\_AddSuggestionType stored

procedure, 385

WROX\_DeleteSuggestion stored

procedure, 380–381

WROX\_DeleteSuggestionType stored

procedure, 385–386

WROX\_GetSuggestion stored

procedure, 381

WROX\_GetSuggestionIDs stored

procedure, 381

WROX\_GetSuggestionModulesAllTabs

stored procedure, 383

WROX\_GetSuggestionModulesThisTab

stored procedure, 383–384

WROX\_GetSuggestions stored

procedure, 381–382

WROX\_GetSuggestionTypes stored

procedure, 386

WROX\_Suggestion table, 378–379

WROX\_SuggestionType table, 384–385

WROX\_UpdateSuggestion stored

procedure, 382

WROX\_UpdateSuggestionType stored

procedure, 386

**WROX.Suggestion module,**

**Presentation Layer**

code-behind regions in, 428–430,

434–436

control code-behind class in, 441–442

control resources of, 439–441,

449–450

CSS styling within user controls in,

422–423

Edit control generally, 446–449

EditSuggestion.aspx in. See

EditSuggestion.aspx

exception handling in, 454–455

helper functions for, 454

introduction to, 419

localization of, 423–424

navigation URLs and, 455–456

optional interfaces for, 426–428

PortalModuleBase class in,

425–426

secondary View control in, 430–434

Settings control for, 436–438

styling, 432–433, 438–439, 449

user controls in, 419–420

View controls in, 421–425

**WSP (Web Site Project)**, 356–358

**www.dotnetnuke.com**, 76

## X

**XmlHttp requests**, 321

**XML/XSL module**, 105, 220–221

**XSLT Generator and Editor**, 215–216

**XXL forks**, 16–17

## Y

**Yahoo!**, 498–500

**YUI Grids, Fonts, and Reset libraries**,

498–500

## Z

**zip files**, 360–361, 462