



# Index

---

## A

### abstraction, 10-15

- definition, 10
  - in APIs, 14
  - in assembly language, 11-12
  - in C language, 13-14
  - GUI application main loops, 309-313
    - CreateUI() function, 312
    - gtk\_init() function, 310
    - gtk\_main() function, 310
  - GUI class, 310-313
  - Init() function, 312
  - MainLoop() function, 312
  - m\_guiImpl member, 311
  - Shutdown() function, 312
  - in IDL (interactive data language), 13
- AC\_MSG\_CHECKING macro, 89
- AC\_MSG\_RESULT macro, 89
- AC\_TRY\_RUN macro, 89

- accessibility, bug reporting and tracking systems, 133
- Add() function, 342-344, 384
- AddAEEncryptClass() function, 512
- AddCatalog() function, 425-426
- addDirectory() function, 169
- addFile() function, 169
- AddProcess() function, 321-322
- AEEncrypt() function, 515
- AEEncrypt\_Decrypt() function, 514
- AEEncrypt\_Encrypt() function, 513
- AEEncrypt\_Get\_key() function, 514
- AEEncrypt\_SetKey() function, 513
- AEEncrypt\_Set\_key() function, 514
- Alexandrescu, Andrei, 31
- alignment of structs, 299-301
- Alphabetical Class Reference (wxWidgets), 331
- ansi argument (gcc), 227

- APIs (application programming interfaces), 221-222. *See also* NSPR (Netscape Portable Runtime Library)
  - advantages, 244
  - abstraction, 14
  - API differences, 243
  - BSD (Berkeley Standard Distribution), 225
  - choosing, 240-241
  - GCC standards support
    - compiler flags, 227
    - Cygwin, 238-240
    - headers, 227-228
    - macros, 228-231
    - MinGW, 234-237
  - GNU C library, 226
  - POSIX, 222-223
    - Microsoft Runtime Library support for, 231-233
    - The Single UNIX Specification, 224-225
    - System V Interface Description (SVID), 223-224
  - XPG (X/Open Portability Guide), 225

- App object, 466-469
  - \$APPDATA variable, 218
  - Append() function, 369
  - AppImpl class, 467
  - Apple Human Interface Guidelines, 323
  - application main loop (Trixul)
    - App object, 466-469
    - AppImpl class, 467
    - in Cocoa, 469
    - in Gtk+, 470
    - in Windows, 470-471
  - application programming interfaces *See* APIs
  - applications
    - adding to Start menu, 178-181
    - Hello World, 335
      - Add() function, 342-344
    - building on Linux, 347-348
    - building on Mac OS X, 348
    - building on Windows, 348-349
    - container widgets, 336-337
    - control widgets, 336
    - main window, 338-345
    - MyFrame class, 338-339
    - OnInit() function, 338
    - portability, 1-3
    - source code listing, 345-346
    - wxBoxSizer class, 341-345
    - wxFrame class, 339
    - wxStaticText class, 340-341
  - Mac OS X application layout, 196-202
  - MIME types, setting, 181
    - restricting for use with GNOME or KDE, 181
    - wxWidget. *See* wxWidgets
  - Aqua, 304, 323
  - arg element, 499
  - The Art of UNIX Programming*, 158
  - assembly language, 11-12
  - atof() function, 227-228
  - autoconf utility, 87-90
  - automake utility, 87-90
  - availablePrinters() function, 445
- B**
- Backprop, 181-182, 185
  - bar program
    - bar.cpp file, 78, 113
    - bar.h file, 77, 112
    - building with Imake utility, 95-98
    - building with make utility, 78-81
    - main.cpp file, 77
  - base OS service routines (POSIX), 223
  - BEGIN\_EVENT\_TABLE macro, 352-353
  - Berkeley Standard Distribution (BSD), 225-226
  - binary data, 280
    - binary socakaddr\_in file, generating, 283-293
    - command line arguments, 283
    - compiler options, 292-293
    - cross-platform solutions, 288-293
    - DumpAddr() function, 289
    - htons() function, 288
    - ntohs() function, 289
    - on Linux, 287
    - on Mac OS X, 287-288
    - sin\_family field, 287
    - sin\_len field, 287
    - sin\_port field, 288
    - sockaddr\_in struct, 290
    - source code listing, 283-287
    - XPBinaryRead() function, 290-292
    - XPBinaryWrite() function, 290-292
  - effect on software portability, 5
  - endian issues, 281
  - intrinsic types and enums, 281
  - NSPR operators, 248-249
  - serializing floating-point numbers as, 276-277
  - struct layout, 281
  - type definitions, 281-282
  - Blanchette, Jasmine, 331
  - Blit() function, 376-377
  - Bluecurve, 179
  - body element (HTML), 430
  - Boost, 6, 263-266
  - boxes
    - Box class, 448
    - box element, 478
    - XUL, 439-441
  - BSD (Berkeley Standard Distribution), 225-226
    - \_BSD\_SOURCE macro, 230
  - bug reporting and tracking systems
    - accessibility, 133
    - Bugzilla, 133-140
      - filing bug reports, 138
      - installing, 134-135
      - main screen, 135
      - Platforms screen, 137

- Products screen, 136
    - searching for bugs, 138-140
    - platform-specific bugs, tracking, 133
  - Bugzilla, 133-140**
    - filing bug reports, 138
    - installing, 134-135
    - main screen, 135
    - Platforms screen, 137
    - Products screen, 136
    - searching for bugs, 138-140
  - build systems, effect on software portability, 9**
  - build.sh script, 78-80**
  - building code, 49-52, 76**
    - autoconf/automake, 87-90
  - Imake
    - bar program example, 95-98
    - building debug, 130
    - building projects with subdirectories, 108-130
    - darwin.cf file, 107
    - darwin.rules file, 107
    - eliminating #ifdefs in code, 101-106
    - Hello World example, 93-94
    - Imake.rules file, 107
    - Imake.tpl file, 107
    - Makefiles, 94-95, 108
    - installing on Mac OS X, 91
    - installing on Windows, 91-93
    - linux.cf file, 107
    - Makefiles, 108
    - overriding defaults with site.def, 99-101
    - Site.def file, 108
    - Win32.cf file, 107
    - Win32.rules file, 107
  - make utility
    - bar program example, 77-78
    - bar.cpp file, 78
    - bar.h file, 77
    - main.cpp file, 77
    - compared to build.sh script, 78-80
    - Makefiles, 79-81
  - on Windows
    - conditional compilation, 84-85
    - nmake utility, 81-84
    - separate source files, 85-87
  - wxWidget
    - applications, 345
    - on Linux, 347-348
    - on Mac OS X, 348
    - on Windows, 348-349
  - bundle icon files, creating, 207-208**
  - Button class**
    - implementation classes
      - ButtonImpl, 453-456
      - CocoaButtonImpl, 456-457
      - GtkButtonImpl, 457-458
      - WidgetImpl, 454-455
      - WindowsButtonImpl, 458-459
    - instantiating, 449-452
  - button element (XML), 440**
  - ButtonImpl class, 453-456**
  - buttons**
    - Button class
      - implementation classes, 453-459
      - instantiating, 449-452
    - button element (XML), 440
    - ButtonImpl class, 453-456
  - Click Me button, creating, 354-355
  - Enable/Disable button, creating, 356-359
- ## C
- C language**
    - abstraction, 13-14
    - portability, 3
  - C++ integration with Trixul, 496-497**
    - accessing components
      - from JavaScript, 500-502
    - coding C++ components, 502-507
    - creating base JavaScript objects, 507-508
    - defining JavaScript functions, 508-509
    - describing components in XML, 498-500
    - generating source code from SIL files, 509-515
    - instantiating C++ objects, 508
    - linking and distributing components, 517
    - resolving points to creator/destroyer functions, 507
    - step-by-step sequence, 497-498
    - writing C++ code to implement components, 515-517
  - C++ GUI Programming with Qt 3, 331**
  - C++ GUI Programming with Qt 4, 324, 332**
  - The C++ Programming Language, 299**
  - CaesarEncrypt() function, 490

- cancelInstall() function, 169
- canonical data, 248
- Cascading Style Sheets (CSS), 437-438
- CDE (Common Desktop Environment), 323
- CDEBUGFLAGS macro, 130
- CFAppleHelpAnchor key (Mac OS X), 200
- CFBundleDevelopmentRegion key (Mac OS X), 201
- CFBundleDocumentTypes key (Mac OS X), 201
- CFBundleExecutable key (Mac OS X), 201
- CFBundleGetInfoString key (Mac OS X), 201
- CFBundleIconFile key (Mac OS X), 201, 207
- CFBundleIdentifier key (Mac OS X), 201
- CFBundleInfoDictionaryVersion key (Mac OS X), 201
- CFBundleName key (Mac OS X), 201
- CFBundlePackageType key (Mac OS X), 201
- CFBundleShortVersionString key (Mac OS X), 201
- CFBundleSignature key (Mac OS X), 202
- CFBundleURLTypes key (Mac OS X), 202
- char types, signed versus unsigned, 278-280
- CheckBox class, 448
- checkbox element (XML), 441
- checksetup command, 134
- choosing
  - compilers, 66-67
  - standards, 240-241
- chrome, 169
- class element, 498
- classes
  - AppImpl, 467
  - Box, 448
  - Button
    - implementation, 453-459
    - instantiating, 449-452
  - ButtonImpl, 453-456
  - CheckBox, 448
  - CocoaButtonImpl, 456-457
  - CocoaFactory, 465-466
  - CocoaGUIImpl, 315
  - CocoaProcessesImpl, 38-42
  - ColorDialog, 400-404
  - Control, 450-451
  - DrawingArea, 379-384, 391
  - fstream, 266
  - GridList, 447
  - GtkButtonImpl, 457-458
  - GUI, 310-313
  - GUIImpl, 314
  - LinuxFactory, 307-308
  - LinuxGUIImpl, 308, 317
  - LinuxProcessesImpl, 35-38
  - Menu, 447
  - MenuBar, 447
  - MenuItem, 447
  - MFC (Microsoft Foundation Classes), 324
  - MyFrame, 338-339
  - OpenPicker, 447
  - PasswordEntryDialog, 395-398
  - printsettings-service, 445
  - ProcessesFactory, 28-29, 306-307
  - ProcessesImpl
    - base class, 31
    - CocoaProcessesImpl class, 38-42
    - LinuxProcessesImpl class, 35-38
    - WindowsProcessesImpl class, 32-35
  - ProcessList, 23-29, 42-45
  - ProcessListener, 319
  - RadioButton, 448
  - SaveAsPicker, 448
  - ScrolledView, 448
  - ScrolledWindow, 448
  - Spacer, 448
  - StaticText, 448
  - Text, 448
  - TopLevel, 364
  - WidgetFactory, 463-465
  - WidgetImpl, 454-455
  - Window, 448
  - WindowsButtonImpl, 458-459
  - WindowsFactory, 29-31
  - WindowsGUIImpl, 316
  - WindowsProcessesImpl, 32-35
  - wxApp, 337-338
  - wxBoxSizer, 336, 341-345, 378, 384-388
  - wxBrush, 374
  - wxColour, 374
  - wxColourDialog, 409-410
  - wxEvtHandler, 336
  - wxFileDialog, 406-408
  - wxFontDialog, 404-405
  - wxFrame, 336, 339, 364-366
    - constructor, 364-365
    - sample implementation, 364-367
  - wxGridSizer, 378
  - wxLocale, 421-422
  - wxMemoryDC, 375
  - wxMenu, 368-369
  - wxMenuBar, 369-370
  - wxMessageBox, 393-394
  - wxNotebookSizer, 378

- wxPaint, 373
- wxPaintDC, 373-375
- wxSlider, 387
- wxStaticBoxSizer, 378
- wxStaticText, 336, 340-341
- wxTextEntryDialog, 395
- wxWindow, 336
- Click Me button, creating, 354-355**
- Cocoa, 304, 324**
  - CocoaButtonImpl class, 456-457
  - CocoaFactory class, 465-466
  - CocoaGUIImpl class, 315
  - Trixul application main loops, 469
  - Trixul implementation objects, 460-461
- CocoaButtonImpl class, 456-457**
- CocoaFactory class, 465-466**
- CocoaGUIImpl class, 315**
- CocoaProcessesImpl class, 38-42**
- CodeWarrior IDE, 67**
- ColorDialog class, 400-404**
- command-line utilities**
  - autoconf, 87-90
  - automake, 87-90
  - checksetup, 134
  - cp, 75
  - CreateShortCut, 217
  - cvs
    - cvs add, 157
    - cvs checkout, 154
    - cvs commit, 155-157
    - cvs diff, 150-151
    - cvs init, 154
  - echo, 75
  - Imake
    - bar program example, 95-98
    - building debug, 130
  - building projects
    - with subdirectories, 108-130
  - darwin.cf file, 107
  - darwin.rules file, 107
  - eliminating #ifdefs in code, 101-106
  - Hello World example, 93-94
  - Imake.rules file, 107
  - Imake.tpl file, 107
  - Imakefiles, 94-95, 108
  - installing on Mac OS X, 91
  - installing on Windows, 91-93
  - linux.cf file, 107
  - Makefiles, 108
  - overriding defaults with site.def, 99-101
  - Site.def file, 108
  - Win32.cf file, 107
  - Win32.rules file, 107
- listdlls, 236
- ls, 172
- make
  - bar program example, 77-78
  - compared to build.sh script, 78-80
  - Makefiles, 79-81
- makensis, 214
- man, 186-189
- mingwtest, 235-236
- mkdir, 75
- msgfmt, 419
- nmake, 81-84
- nroff, 192
- otool, 203
- open, 185
- sed, 75
- uname, 50
- UninstPage, 216
- view, 185
- wx-config, 333
- xmkmf, 96-98
- Common Desktop Environment (CDE), 323**
- Common Software Environment (COSE), 323**
- compatibility files, 105-106**
- compatibility libraries, Imakefiles, 116-119**
- compilers**
  - building source code with multiple compilers, 52-56
  - choosing, 66-67
  - compiler differences, 242-243
  - conditional compilation, 84-85
  - effect on software portability, 3-4
  - g++, 280
  - GCC
    - compiler flags, 227
    - headers, 227-228
    - macros, 228-231
    - MinGW, 234-240
  - warnings, 61-62
    - GNU flags, 62-63
    - Microsoft Visual C++ flags, 63-64
- compiling code. *See also* building code**
  - conditional compilation, 84-85
  - NSIS script, 214
- ComplexProgramTarget macro, 116**
- component element, 498**
- composite widgets (wxWidget)**
  - wxColourDialog class, 409-410
  - wxFileDialog class, 406-408
  - wxFontDialog class, 404-405
- Concurrent Version System. *See* CVS**

- conditional compilation, 84-85
  - configuration
    - CVS (Concurrent Version System), 152-156
    - software configuration management. *See* SCM
  - constants, XP\_PUTENV, 102
  - container widgets (wxWidget), 363-364, 450
    - responding to user input, 388-391
      - EVT\_CHECKBOX macros, 389-390
      - EVT\_COMMAND\_SCROLL macro, 388
      - GetRadiusPercent() function, 388
      - IsChecked() function, 390
      - OnSliderChange() function, 388
      - SetRadiusPercent() function, 388
    - wxBoxSizer class, 378, 384-388
    - wxBrush class, 374
    - wxColour class, 374
    - wxFrame class
      - constructor, 364-365
      - sample implementation, 364-367
    - wxGridSizer class, 378
    - wxMemoryDC class, 375
    - wxMenu class, 368-369
    - wxMenuBar class, 369-370
    - wxNotebookSizer class, 378
    - wxPaint class, 373
    - wxPaintDC class, 373-375
    - wxSlider class, 387
    - wxStaticBoxSizer class, 378
  - Control class, 450-451
  - control widgets, 450
  - controls (XUL), 441
  - conversion operators (NSPR), 247
  - converting integers, 298-299
  - COSE (Common Software Environment), 323
  - cp command, 75
  - CppCmd macro, 93
  - cppflags argument (wx-config), 333
  - CppFileTarget macro, 126
  - creat() function, 232
  - Create() function, 449, 459, 475-476
  - CreateChildren() function, 476
  - CreateFile() function, 232
  - CreateJSObject() function, 500, 504-506
  - CreateMenus() function, 317-318
  - CreateProcess() function, 6
  - CreateShortCut command, 217
  - CreateThread() function, 251-254
  - CreateUI() function, 312
  - Cross-Platform GUI Programming with wxWidgets*, 331
  - cross-platform GUI toolkits, 326, 427-428. *See also* Trixul; XUL (XML User Interface Language)
  - CSS (Cascading Style Sheets), 437-438
  - custom GUI toolkits, 327. *See also* Trixul
  - CVS (Concurrent Version System), 10, 149-157
    - commands
      - cvs add, 157
      - cvs checkout, 154
      - cvs commit, 155-157
      - cvs diff, 150-151
      - cvs init, 154
    - organizing projects in, 45-49
    - running, 157
    - setting up, 152-156
  - CVSROOT directory, 154-156
  - Cygwin, 71-76
    - downloading and installing, 238-239
    - Imake for GCC, installing, 91-92
    - testing, 239-240
  - cygwintest.cpp file, 239-240
- D**
- darwin.cf file, 107
  - darwin.rules file, 107
  - data serialization, binary data, 280
    - binary socakddr\_in file, generating, 283-293
    - endian issues, 281
    - intrinsic types and enums, 281
    - struct layout, 281
    - type definitions, 281-282
  - data types
    - char, 278-280
    - DIR, 229-230
    - integers, 298-299
    - NSPR types, 245-246
    - sizes
      - efficiency, 297-298
      - integer conversions, 298-299
      - integer types, 293-296

- Netscape Portable Runtime Library (NSPR), 296-297
  - struct alignment and ordering, 299-301
  - debuggers**
    - GNU debugger (gdb), 69
    - Xcode, 68
  - DECLARE\_EVENT\_TABLE\_ENTRY** macro, 361
  - DECLARE\_EVENT\_TABLE** macro, 350-351, 371
  - Decrypt()** function, 511, 516
  - defaults (Imake), overriding with site.def, 99-101
  - DeleteAesEncryptProxy()** function, 513
  - DeleteClassProxy()** function, 507
  - design, Observer (publish/subscribe) design pattern, 318-322
  - desktops. *See* GNOME; KDE
  - Destroy()** function, 403
  - DHTML (Dynamic HTML)**
    - CSS (Cascading Style Sheets), 437-438
    - DOM (Document Object Model), 434-437
    - HTML (Hypertext Markup Language), 429-432
      - elements, 430-432
      - Hello World! example, 430
    - scripting language, 433-434
  - dialog element (XML)**, 439
  - dialogs (wxWidget)**
    - modal dialogs, 392-394
      - creating with wxMessageBox, 393-394
    - custom dialogs, 394-398
  - definition of, 392
  - PasswordEntryDialog** class, 395-398
  - wxTextEntryDialog** class, 395
  - modeless dialogs, 392, 399-404
  - wxColourDialog** class, 409-410
  - wxFileDialog** class, 406-408
  - wxFontDialog** class, 404-405
  - diff** files, creating, 150-151
  - DIR** type, 229-230
  - directories**
    - CVSROOT, 154-156
    - dist, 74
    - inc, 47
    - lib, 46
    - main, 48
    - Resources (Mac OS X), 204-205
    - /usr/local/man, 195
  - Disable()** function, 453
  - Disable/Enable** button, creating, 356-359
  - disabling controls**, 450
  - dist** directory, 74
  - distributing**
    - components, 517
    - man pages, 195-196
    - MO files, 421
    - NSIS installer, 214
  - dlopen()** function, 261
  - Document object**, 472-473
  - Document Object Model (DOM)**, 434-437
    - interacting with Trixul widgets from JavaScript, 488-493
    - mapping DOM objects to JavaScript, 493-496
    - XUL, 443-444
  - documentation (wxWidgets)**, 331
  - documents (Trixul)**, 472-473
  - DOM (Document Object Model)**, 434-437
    - interacting with Trixul widgets from JavaScript, 488-493
    - mapping DOM objects to JavaScript, 493-496
    - XUL, 443-444
  - downloading**
    - Cygwin, 238-239
    - MinGW, 234
    - wxWidgets, 331
  - drag-and-drop installation (Mac OS X)**, 208
  - DrawCircle()** function, 374
  - drawing content in frames**
    - Blit() function, 376-377
    - DrawCircle() function, 374
    - GetClientSize() function, 375
    - OnPaint() function, 373-378
    - SetBrush() function, 374
    - wxPaint class, 373
    - wxPaintDC class, 373
  - DrawingArea** class, 379-384, 391
  - dump()** function, 487
  - DumpAddr()** function, 289
  - DXP\_PUTENV** macro, 85
  - Dynamic HTML.** *See* DHTML
- E**
- echo command, 75
  - ECMA (European Computer Manufacturers Association)**, 433
  - ECMAScript**, 433
  - Effective C++*, 63, 266
  - Element object**, 473-475

- elements, 430-432
    - arg, 499
    - body, 430
    - box, 478
    - checkbox, 441
    - class, 498
    - component, 498
    - Element object, 473-475
    - function, 499
    - head, 430
    - html, 430
    - image, 432
    - label, 442
    - li, 431
    - link, 438
    - progressmeter, 442
    - property, 499
    - radio, 441
    - return, 499
    - script, 433
    - scrolledview, 484-485
    - scrolledwindow, 484-485
    - spacer, 480
    - title, 430
    - ul, 431
    - window, 439
  - Enable() function, 358, 453
  - Enable/Disable button,
    - creating, 356-359
  - Encrypt() function, 508-511, 515
  - endian systems, binary data issues, 281
  - EndModal() function, 398
  - END\_EVENT\_TABLE
    - macro, 352-353
  - enums
    - binary data issues, 281
    - wxLanguage, 422-424
  - environment variables,
    - LD\_LIBRARY\_PATH, 173-176
  - equality of floating-point numbers, 277-278
  - /etc/gnome-vfs-mime-magic file, 183
  - /etc/group file, 171
  - /etc/mime-magic file, 183
  - /etc/password file, 171
  - European Computer Manufacturers Association (ECMA), 433
  - event handler functions.
    - See functions
  - events (wxWidget), 349
    - BEGIN\_EVENT\_TABLE
      - macro, 352-353
    - Click Me button,
      - creating, 354-355
    - DECLARE\_EVENT\_TAB
      - LE macro, 350-351
    - Enable/Disable button,
      - creating, 356-359
    - END\_EVENT\_TABLE
      - macro, 352-353
    - event handler function
      - implementations, 355
    - EVT\_BUTTON
      - macro, 353
    - EVT\_MENU macro, 353
    - File menu, 354
    - skipping, 359-363
    - wxCommandEvent
      - events, 351
  - EVT\_BUTTON macro, 353
  - EVT\_CHECKBOX macros, 389-390
  - EVT\_CLOSE macro, 361
  - EVT\_COMMAND\_SCROLL
    - L macro, 388
  - EVT\_MENU macro, 353
  - EVT\_PAINT() macro, 373
  - execute (x) permissions, 172
  - expat shared library, 203
- F**
- factories, 29, 463-466
    - CocoaFactory class, 465-466
    - GetWidgetFactory()
      - function, 464
    - ProcessesFactory class, 28-29
    - WidgetFactory class, 463-465
  - fields
    - sin\_family, 287
    - sin\_len, 287
    - sin\_port, 288
  - File menus, creating, 354
  - files
    - associating with
      - icons, 183-186
    - bar program
      - bar.cpp, 78, 113
      - bar.h, 77, 112
      - main.cpp, 77
    - binary sockaddr\_in file,
      - generating, 283-293
    - command-line
      - arguments, 283
    - compiler options, 292-293
    - cross-platform
      - solutions, 288-293
    - DumpAddr()
      - function, 289
    - htons() function, 288
    - ntohs() function, 289
    - on Linux, 287
    - on Mac OS X, 287-288
    - sin\_family field, 287
    - sin\_len field, 287
    - sin\_port field, 288
    - sockaddr\_in
      - struct, 290
    - source code listing, 283-287
    - XPBinaryRead()
      - function, 290-292
    - XPBinaryWrite()
      - function, 290-292
  - bundle icon files, creating, 207-208
  - cygwintest.cpp, 239-240



- determining file types, 182-183
- diff files, creating, 150-151
- /etc/gnome-vfs-mime-magic, 183
- /etc/group, 171
- /etc/mime-magic, 183
- /etc/password, 171
- gnumeric.keys, 184
- Hello World program
  - hello.cpp, 93
  - hello.h, 93
  - Imakefiles, 94
- Imake
  - building debug, 130
  - darwin.cf, 107
  - darwin.rules, 107
  - Imake.rules, 107
  - Imake.tmpl, 107
  - Imakefiles, 108
  - linux.cf, 107
  - Makefiles, 108
  - Site.def, 108
  - Win32.cf, 107
  - Win32.rules, 107
- Info.plist, 199-200
- install.js, 167-169
- Localizable.strings, 205-206
- Makefiles, 49-52, 79-81
- message files, loading at runtime, 425-426
- MO files
  - distributing, 421
  - generating, 419
  - locating, 420-421
- per-platform compatibility files, 105-106
- permissions, 171-173
- PkgInfo, 198
- PO files
  - generating, 417-419
  - locating, 420-421
- reject files (patch program), 162-163
- SIL files, 509-515
- version.plist, 197-198
- XPI files, 167
- filing bug reports (Bugzilla), 138**
- finding bugs (Bugzilla), 138-140**
- flags. *See specific flags***
- floating-point numbers**
  - equality, 277-278
  - serializing as binary, 276-277
  - support for IEEE-754 standard, 274-275
- folders, MacOS, 202-206**
- frames**
  - drawing content
    - in, 373-378
    - Blit() function, 376-377
    - DrawCircle() function, 374
    - GetClientSize() function, 375
    - OnPaint() function, 373-378
    - SetBrush() function, 374
    - wxPaint class, 373
    - wxPaintDC class, 373
    - wxFrame class, 364
- free() function, 271**
- fstream class, 266**
- function element, 499**
- FunctionName() function, 371**
- functions. *See also macros***
  - Add(), 342-344, 384
  - AddAEEncryptClass(), 512
  - AddCatalog(), 425-426
  - addDirectory(), 169
  - addFile(), 169
  - AddProcess(), 321-322
  - AEEncrypt(), 515
  - AEEncrypt\_  
Decrypt(), 514
  - AEEncrypt\_  
Encrypt(), 513
  - AEEncrypt\_  
Get\_key(), 514
  - AEEncrypt\_  
SetKey(), 513
  - AEEncrypt\_  
Set\_key(), 514
  - Append(), 369
  - associating with menu items, 370-372
  - atof(), 227-228
  - availablePrinters(), 445
  - Blit(), 376-377
  - CaesarEncrypt(), 490
  - cancelInstall(), 169
  - creat(), 232
  - Create(), 449, 459, 475-476
  - CreateChildren(), 476
  - CreateFile(), 232
  - CreateJSObject(), 500, 504-506
  - CreateMenu(), 317-318
  - CreateProcess(), 6
  - CreateThread(), 251
  - CreateThread(), 252-254
  - CreateUI(), 312
  - Decrypt(), 511, 516
  - defining, 508-509
  - DeleteAEEncrypt-Proxy(), 513
  - DeleteClassProxy(), 507
  - Destroy(), 403
  - Disable(), 453
  - dlopen(), 261
  - DrawCircle(), 374
  - dump(), 487
  - DumpAddr(), 289
  - Enable(), 358, 453
  - Encrypt(), 508-511, 515
  - EndModal(), 398
  - free(), 271

- FunctionName(), 371
- GetAESEncrypt(), 509
- GetAESEncryptClass(), 512
- GetAlignment(), 150
- GetAppInstance(), 467
- GetClientSize(), 375
- GetColourData(), 410
- GetComponent(), 501
- GetCount(), 24
- getElementById(), 434, 437, 443, 473, 493-495
- GetFactoryInstance(), 29-31, 43, 466
- GetFilename(), 408
- GetFileNames(), 408
- GetLabel(), 451-453
- GetMatchingFiles(), 268
- GetName(), 24
- getObject(), 501-504
- GetPaths(), 408
- GetPID(), 24
- GetProcessesFactory(), 28, 43
- GetRadiusPercent(), 388
- GetUsername(), 397
- GetWidgetFactory(), 464-466
- Get\_key(), 511
- GPGAESEncrypt(), 501-502
- gtk\_button\_new(), 452
- gtk\_init(), 310
- gtk\_main(), 310
- HandleEncrypt(), 490-492
- Hide(), 449, 453
- htons(), 288
- Init(), 312, 421
- Initialize(), 467-468
- initInstall(), 169
- IsChecked(), 390
- JS\_DefineObject(), 507-508
- LoadFile(), 270-271
- LoadLibrary(), 261
- MainLoop(), 312, 467-468
- MakeButton(), 466
- MakeGUI(), 307
- MakeProcesses(), 30
- malloc(), 271
- MyThreadFunc(), 254
- NewAESEncryptProxy(), 513
- NewClassProxy(), 507
- nice(), 27
- NSPR standard library (libc) functions, 260
- I/O functions, 261
- linking functions, 261-263
- NtCreateFile(), 232
- ntohs(), 289
- OnApply(), 401
- OnBlue(), 401
- OnCancel(), 397
- OnClickHandler(), 487
- OnClickMe(), 355
- OnClose(), 361-362
- OnColor(), 409
- OnDisableEnable(), 359
- OnFile(), 408
- OnFont(), 405
- OnGreen(), 401
- OnInit(), 337
- OnNewProcess(), 319-321
- OnOK(), 397, 474
- OnPaint(), 373-378
- OnPassword(), 396
- OnQuit(), 355, 359
- OnRed(), 401
- OnSliderChange(), 388
- open(), 233
- opendir(), 236
- performInstall(), 169
- PL\_CreateOptState(), 261
- PL\_GetNextOpt(), 261
- PL\_strdup(), 261
- PL\_strlen(), 261
- PL\_strncasestr(), 261
- PL\_strncpy(), 261
- PL\_strrstr(), 261
- PortableOpen(), 14
- POSIX SVID (System V Interface Description)
  - base OS service routines, 223
  - general library functions, 224
  - mathematical functions, 223
  - networking functions, 224
  - string functions, 224
- PR\_CreateThread(), 258-259
- PR\_htonl(), 249
- PR\_htons(), 249
- PR\_LoadLibrary(), 261-263
- PR\_ntohl(), 249
- PR\_ntohs(), 249
- PR\_OpenDir(), 269
- PR\_ReadDir(), 269
- PrintProcessName
  - AndID(), 33
- pthread\_attr\_init(), 257
- pthread\_attr\_setstacksize(), 257
- pthread\_create(), 254-256
- pthread\_exit(), 256-257
- pthread\_join(), 256-257
- PutBar(), 78, 85, 100
- putenv(), 84, 102-105
- QueryInterface(), 446
- quit(), 487
- Refresh(), 401
- regcomp(), 269
- regex, 265-268
- regexexec(), 269-270
- registerChrome(), 169
- RegisterListener(), 320
- releaseObject(), 501

- RemoveAEEncryptClass(), 512
  - rename(), 241
  - ReverseValue(), 443-444
  - Scan(), 23, 26-27, 31-33, 39, 43-45, 313
  - ScanProcesses(), 33-34
  - SendNotification(), 320-321
  - SetBar(), 78
  - SetBrush(), 374
  - setenv(), 84
  - SetGeometry(), 449
  - SetKey(), 511, 515
  - SetLabel(), 451-453
  - SetMenuBar(), 354, 369
  - SetPriority(), 27
  - SetPriorityClass(), 27
  - SetRadiusPercent(), 388
  - SetSizer(), 355, 385
  - SetTopWindow(), 367
  - Set\_key(), 511
  - Show(), 367, 402-403, 449, 453
  - ShowModal(), 395-398, 407
  - Shutdown(), 312, 467-468
  - sizeof(), 294
  - Skip(), 362
  - snprintf(), 276
  - SomeFuncInt(), 279
  - SomeFuncUInt(), 279
  - strftime(), 88-89
  - strtod(), 276
  - sysctl(), 38
  - UnregisterListener(), 320
  - UppercaseValue(), 443-444
  - wxGetTranslation(), 417
  - XPBinaryRead(), 290-292
  - XPBinaryWrite(), 290-292
- G**
- g++ compiler, 280
  - GCC
    - compiler flags, 227
    - Cygwin
      - downloading and installing, 238-239
      - testing, 239-240
    - headers, 227-228
    - Imake for GCC, installing, 91-92
    - macros, 228-231
    - MinGW
      - definition of, 234
      - downloading and installing, 234
      - testing, 235-236
      - verifying libraries, 236-237
  - gdb (GNU debugger), 69
  - general library functions (POSIX), 224
  - Get\_key() function, 511
  - GetAEEncrypt() function, 509
  - GetAEEncryptClass() function, 512
  - GetAlignment() function, 150
  - GetAppInstance() function, 467
  - GetClientSize() function, 375
  - GetColourData() function, 410
  - GetComponent() function, 501
  - GetCount() function, 24
  - getElementById() function, 434, 437, 443, 473, 493-495
  - GetFactoryInstance() function, 29-31, 43, 466
  - GetFilename() function, 408
  - GetFileNames() function, 408
  - GetLabel() function, 451-453
  - GetMatchingFiles() function, 268
  - GetName() function, 24
  - getObject() function, 501-504
  - GetPaths() function, 408
  - GetPID() function, 24
  - GetProcessesFactory() function, 28, 43
  - GetRadiusPercent() function, 388
  - gettext, 411-420
  - GetUsername() function, 397
  - GetWidgetFactory() function, 464-466
  - GNOME
    - integrating with Linux installer, 177-178
      - adding applications to Start menu, 178-181
      - associating files with icons, 183-186
      - determining file types, 182-183
      - restricting applications for use with GNOME or KDE, 181
      - setting application MIME types, 181
    - overview, 304, 324
  - GNU
    - C library, 226
    - compiler warnings, 62-63
    - debugger (gdb), 69
  - \_GNU\_SOURCE macro, 231
  - gnumeric.keys file, 184
  - GPAAEEncrypt() function, 501-502
  - graphical user interfaces. *See* GUIs

GridList class, 447

## Gtk+

- GtkButtonImpl class, 457-458
- LinuxGUIImpl class, 317
- Trixul application main loops, 470
- Trixul implementation objects, 461

## Gtk+ Programming in

C, 324

gtk\_button\_new()

function, 452

gtk\_init() function, 310

gtk\_main() function, 310

GtkButtonImpl class, 457-458

GUI class, 310-313

GUIImpl class, 314

GUIs (graphical user interfaces), 303-304

effect on software portability, 8-9

look and feel

standards, 323

model/view paradigm, 305-306

Observer (publish/subscribe) design pattern, 318-322

view implementation, 306, 309-318

themes, 304-305

toolkits, 324-325. *See also specific toolkits*

cross-platform GUI toolkits, 326, 427-428

custom GUI

toolkits, 327

native GUI toolkits, 325-326

## H

HandleEncrypt() function, 490-492

HasPutenv macro, 98

hbox element (XML), 439-440

head element (HTML), 430

headers (GCC), 227-228

Hello World program

(wxWidgets), 335

Add() function, 342-344

building on Linux, 347-348

building on Mac

OS X, 348

building on Windows, 348-349

building with Imake, 93-94

container widgets, 336-337

control widgets, 336

main window, 338-345

MyFrame class, 338-339

OnInit() function, 338

portability, 1, 3

source code listing, 345-346

wxBoxSizer class, 341-345

wxFrame class, 339

wxStaticText class, 340-341

Hello World! Web page, 430

hello.cpp file, 93

hello.h file, 93

Hide() function, 449, 453

history of wxWidgets, 331

HTML (Hypertext Markup Language), 429-432. *See also* DHTML (Dynamic HTML)

elements, 430-432

Hello World!

example, 430

html element (HTML), 430

htons() function, 288

Hypertext Markup

Language. *See* HTML

## I

I/O functions (NSPR), 261

icns files, creating, 207-208

icons, associating files

with, 183-186

id argument (wxFrame), 365

IDEs

Metrowerks CodeWarrior IDE, 67

native IDEs, 67-71

Xcode IDE, 68

IDL (interactive data

language), 13

IEEE-754 standard, 274-275

#ifdefs, eliminating, 101

per-platform

compatibility files, 105-106

putenv function, 102-105

IHaveSubdirs macro, 114

image element (HTML), 432

Imake utility

bar program

example, 95-98

building debug, 130

building projects with

subdirectories, 108-111

compatibility library

Imakefiles, 116-119

Makefiles, 119-130

Windows

implementations, 111-116

darwin.cf file, 107

darwin.rules file, 107

eliminating #ifdefs in code, 101

per-platform

compatibility files, 105-106

putenv() function, 102-105

- Hello World example, 93-94
    - hello.cpp file, 93
    - hello.h file, 93
    - Imakefiles, 94
  - Imake.rules file, 107
  - Imake.tmpl file, 107
  - Imakefiles, 94-95, 108
  - installing on Mac OS X, 91
  - installing on Windows
    - Imake for GCC, 91-92
    - Imake for Visual C++, 92-93
  - linux.cf file, 107
  - Makefiles, 108
  - overriding defaults with site.def, 99-101
  - Site.def file, 108
  - Win32.cf file, 107
  - Win32.rules file, 107
  - Imake.rules file**, 107
  - Imake.tmpl file**, 107
  - Imakefiles**, 94-95, 108
  - IMPLEMENT\_APP macro**, 338
  - inc directory**, 47
  - Info.plist file**, 199-200
  - Init() function**, 312, 421
  - Initialize() function**, 467-468
  - initInstall() function**, 169
  - input, responding to**, 388-391
    - EVT\_CHECKBOX macros, 389-390
    - EVT\_COMMAND\_SCROLL macro, 388
    - GetRadiusPercent() function, 388
    - IsChecked() function, 390
    - OnSliderChange() function, 388
    - SetRadiusPercent() function, 388
  - install.js file**, 167-169
  - installation (software)**, 165-166
    - Cygwin, 76, 238-239
    - Linux platform installs, 170-173
      - /etc/group file, 171
      - /etc/password file, 171
    - executing as root, 176-177
    - execution environment, 173-177
    - file permissions, 171-173
    - integrating with GNOME and KDE desktops, 177-186
    - man pages, 186-196
    - Mac OS X platform installs
      - application layout, 196-202
      - bundle icon files, creating, 207-208
      - drag-and-drop installation, 208
      - MacOS folder, 202-206
    - MinGW, 234
    - Windows XP NSIS installer
      - application data, 217-218
      - compiling NSIS script, 214
      - creating, 210
      - defining, 210-214
      - distributing, 214
      - documentation, 208
      - documents and settings, 208-209
    - integrating into Start menu and desktop, 216-217
    - program installation, 210
    - testing, 214
    - uninstallers, 215-216
  - Weather Manager sample installer script, 218-220
  - XPIInstall, 166-170
  - wxWidgets, 332
    - on Linux, 334
    - on Mac OS X, 333
    - on Windows, 334-335
  - instance hierarchies, creating**, 42-45
  - instantiating C++ objects**, 508
  - integers**
    - conversions, 298-299
    - m\_bar, 78
    - sizes, 293-296
  - interactive data language (IDL)**, 13
  - internationalization (wxWidgets)**, 410-411
    - gettext, 411-420
    - locales, creating, 421-425
    - message files, loading at runtime, 425-426
    - PO/MO files, 417-421
    - wxLanguage enums, 422-424
    - wxLocale class, 421-422
  - intrinsic types**
    - binary data issues, 281
    - type definitions, 281-282
  - IsChecked() function**, 390
  - ISO/IEC 9945-1. See POSIX \_ISOC99\_SOURCE macro**, 230
- ## J
- JavaScript**, 433
    - functions, defining, 508-509
    - integration with Trixul, 485-486
      - example, 486-487
      - including external JavaScript sources, 487-488

- interacting with
  - widgets from JavaScript, 488-493
  - mapping DOM objects to JavaScript, 493-496
  - XUL, 443-444
- JS\_DefineObject()**
  - function, 507-508
- JSClass struct**, 507
- K**
- KDE**
  - integrating with Linux installer, 177-178
    - adding applications to Start menu, 178-181
    - associating files with icons, 183-186
    - determining file types, 182-183
    - restricting applications for use with GNOME or KDE, 181
    - setting application MIME types, 181
  - overview, 304, 324
- L**
- label element (XML)**, 442
- language, effect on software portability**, 3
- layouts (Trixul)**, 477, 480-483
- LD\_LIBRARY\_PATH**
  - environment variable, 173-176
- lexical\_cast**, 277
- li element (HTML)**, 431
- lib directory**, 46
- libraries**
  - effect on software portability, 5-6
  - expat shared library, 203
  - GNU C library, 226
  - Microsoft Runtime Library, 231-233
  - NSPR (Netscape Portable Runtime Library)
    - advantages of, 242-244
    - binary data, 248-249
    - operators, 246-247
    - standard library (libc) functions, 260
    - I/O functions, 261
    - linking functions, 261-263
    - threads. *See* threads
    - types, 245-246, 296-297
    - when to use, 263-271
  - STL (Standard Template Library), 5-6
- libs argument (wx-config)**, 333
- licensing terms**
  - Qt, 331
  - wxWidgets, 331-332
- link element (HTML)**, 438
- LINKFLAGS macro**, 86
- linking functions (NSPR)**, 261-263
- Linux**
  - binary socakddr\_in file, generating, 287
  - Bluecurve, 179
  - building wxWidget applications on, 347-348
  - GNOME, 304, 324
  - GUI view implementation
    - LinuxFactory class, 307-308
    - LinuxGUIImpl class, 308
    - ProcessesFactory class, 306-307
  - installs. *See* Linux installer
  - KDE, 304, 324
  - LinuxGUIImpl GUI implementation class, 317
  - LinuxProcessesImpl class, 35-38
  - threads, 254-257
  - wxWidget
    - installation, 334
- Linux installer, 170-173**
  - /etc/group file, 171
  - /etc/password file, 171
  - executing as root, 176-177
  - execution environment, 173-177
  - file permissions, 171-173
  - integrating with GNOME and KDE desktops, 177-178
    - adding applications to Start menu, 178-181
    - associating files with icons, 183-186
    - determining file types, 182-183
    - restricting applications for use with GNOME or KDE, 181
    - setting application MIME types, 181
  - man pages
    - benefits of, 186-189
    - creating, 189-195
    - distributing, 195-196
- linux.cf file**, 107
- LinuxCompatibility.cpp file**, 106
- LinuxFactory class**, 307-308
- LinuxGUIImpl class**, 308, 317
- LinuxProcessesImpl class**, 35-38
- listdlls command**, 236
- LL\_EQ macro**, 246

- LL\_L2I macro, 247
  - LL\_MAXINT macro, 246
  - LL\_NOT macro, 247
  - LL\_OR macro, 247
  - LL\_SHL macro, 247
  - LL\_XOR macro, 247
  - LoadFile() function, 270-271
  - loading message files at runtime, 425-426
  - LoadLibrary() function, 261
  - locales, creating, 421-425
  - Localizable.strings file, 205-206
  - logical operators (NSPR), 247
  - logicalFunc argument (Blit() function), 376-377
  - look and feel standards (GUIs), 323
  - ls -l command, 172
  - LShasLocalizedDisplayName key (Mac OS X), 202
- M**
- m\_bar integer, 78
  - m\_enabled variable, 358
  - m\_guiImpl member, 311
  - Mac OS X
    - Aqua, 304, 323
    - binary socakddr\_in file, generating, 287-288
    - building wxWidget applications on, 348
    - Cocoa, 324
      - CocoaButtonImpl class, 456-457
      - CocoaFactory class, 465-466
      - Trixul application main loops, 469
      - Trixul implementation objects, 460-461
    - CocoaGUIImpl GUI implementation class, 315
    - CocoaProcessesImpl class, 38-42
    - Gtk+
      - Trixul application main loops, 470
      - Trixul implementation objects, 461
    - Imake installation, 91
    - software installation application layout, 196-202
    - bundle icon files, creating, 207-208
    - drag-and-drop installation, 208
    - MacOS folder, 202-206
    - threads, 254-257
    - Windows
      - Trixul application main loops, 470-471
      - Trixul implementation objects, 462-463
    - wxWidget installation, 333
  - MacOS folder, 202-206
  - macros. *See also* functions
    - AC\_MSG\_CHECKING, 89
    - AC\_MSG\_RESULT, 89
    - AC\_TRY\_RUN, 89
    - BEGIN\_EVENT\_TABLE, 352-353
    - \_BSD\_SOURCE, 230
    - CDEBUGFLAGS, 130
    - ComplexProgramTarget, 116
    - CppCmd, 93
    - CppFileTarget, 126
    - DECLARE\_EVENT\_TABLE\_ENTRY, 361
    - DECLARE\_EVENT\_TABLE, 350-351, 371
    - DXP\_PUTENV, 85
    - END\_EVENT\_TABLE, 352-353
    - EVT\_BUTTON, 353
    - EVT\_CHECKBOX, 389-390
    - EVT\_CLOSE, 361
    - EVT\_COMMAND\_SCROLL, 388
    - EVT\_MENU, 353
    - EVT\_PAINT, 373
    - GCC, 228-231
    - \_GNU\_SOURCE, 231
    - HasPutenv, 98
    - IHaveSubdirs, 114
    - IMPLEMENT\_APP, 338
    - \_ISOC99\_SOURCE, 230
    - LINKFLAGS, 86
    - LL\_EQ, 246
    - LL\_L2I, 247
    - LL\_MAXINT, 246
    - LL\_NOT, 247
    - LL\_OR, 247
    - LL\_SHL, 247
    - LL\_XOR, 247
    - MkdirHierCmd, 93, 120
    - PassCDebugFlags, 130
    - \_POSIX\_C\_SOURCE, 230
    - \_POSIX\_SOURCE, 228-230
    - PR\_INT16\_MAX, 245
    - PR\_INT16\_MIN, 245
    - .SH, 192
    - SharedLibraryTarget, 126
    - \_SVID\_SOURCE, 230
    - .TH, 192
    - \_XOPEN\_SOURCE, 230
  - main directory, 48
  - main.cpp file (bar program), 77
  - MainLoop() function, 312, 467-468

- make systems, 76**
  - autoconf/automake, 87-90
  - building on Windows
    - conditional compilation, 84-85
    - nmake utility, 81-84
    - separate source files, 85-87
- Imake**
  - bar program
    - example, 95-98
  - building debug, 130
  - building projects
    - with subdirectories, 108-130
  - darwin.cf file, 107
  - darwin.rules file, 107
  - eliminating #ifdefs in code, 101-106
  - Hello World
    - example, 93-94
  - Imake.rules file, 107
  - Imake.tmpl file, 107
  - Imakefiles, 94-95, 108
  - installing on Mac OS X, 91
  - installing on Windows, 91-93
  - linux.cf file, 107
  - Makefiles, 108
  - overriding defaults
    - with site.def, 99-101
  - Site.def file, 108
  - Win32.cf file, 107
  - Win32.rules file, 107
- make utility**
  - bar program
    - example, 77-78
  - compared to build.sh script, 78-80
  - Makefiles, 79-81
- make utility**
  - bar program example
    - bar.cpp file, 78
    - bar.h file, 77
    - main.cpp file, 77
  - compared to build.sh script, 78-80
  - Makefiles, 79-81
- MakeButton() function, 466**
- Makefiles, 49-52, 79-81, 108**
  - Imakefiles, 94-95, 108
- MakeGUI() function, 307**
- makensis command, 214**
- MakeProcesses() function, 30**
- malloc() function, 271**
- man command, 186-189**
- man pages**
  - benefits of, 186-189
  - creating, 189-195
  - distributing, 195-196
- management**
  - building source code
    - on multiple platforms, 56-60
  - building source code with multiple compilers, 52-56
  - organizing projects in CVS or SVN, 45-49, 149-151
  - overview, 17
  - parity, 17-21
  - responding to compiler warnings, 61-62
    - GNU flags, 62-63
    - Microsoft Visual C++ flags, 63-64
  - sharing code across all supported platforms
    - abstraction layers, 22
    - advantages, 22
    - implementation
      - classes, 31-32
    - instance hierarchies, 42-45
- Makefiles and building code, 49-52
  - platform factory implementations, 29-31
  - platform-specific ProcessesImpl classes, 32-42
  - ProcessList class example, 23-29
  - software configuration management. *See* SCM
  - testing builds on all supported platforms, 60-61
- mapping DOM objects to JavaScript, 493-496**
- mathematical functions (POSIX), 223**
- mathematical operators (NSPR), 247**
- memory DCs, creating, 375-378**
- menu bars**
  - MenuBar class, 447
  - wxMenuBar class, 369-370
- Menu class, 447**
- MenuBar class, 447**
- MenuItem class, 447**
- menus, 367**
  - associating with functions, 370-372
  - creating with wxMenu class, 368-369
  - menu bars, creating, 369-370, 447
  - XUL, 441-442
- message files, loading at runtime, 425-426**
- Metrowerks CodeWarrior IDE, 67**
- Meyer, Scott, 266**
- MFC (Microsoft Foundation Classes), 324**



- Microsoft Runtime Library, 231-233, 240
  - Microsoft Visual C++, 63-64
  - Microsoft Windows User Experience*, 323
  - MIME types, 181
  - MinGW
    - definition of, 234
    - downloading and installing, 234
    - testing, 235-236
    - verifying libraries, 236-237
  - mingwtest command, 235-236
  - mkdir command, 75
  - MkdirHierCmd macro, 93, 120
  - MO files
    - distributing, 421
    - generating, 419
    - locating, 420-421
  - modal dialogs, 392-394
    - creating with wxMessageBox, 393-394
    - custom dialogs, 394-398
    - definition of, 392
    - PasswordEntryDialog class, 395-398
    - wxTextEntryDialog class, 395
  - model/view paradigm (GUIs), 305-306
    - Observer (publish/subscribe) design pattern, 318-322
    - view implementation
      - abstracting GUI main loop, 309-313
      - CocoaGUIImpl class, 315
      - CreateMenus() function, 317-318
      - GUIImpl class, 314
      - illustration of UI implementation, 306
      - LinuxFactory class, 307-308
      - LinuxGUIImpl class, 308, 317
      - ProcessesFactory class, 306-307
      - WindowsGUIImpl class, 316
  - modeless dialogs, 392, 399-404
  - Modern C++ Design: Generic Programming and Design Patterns Applied*, 31
  - Motif, 324
  - Mozilla
    - chrome, 169
    - Mozilla LXR, 167
    - XPCOM, 444-446
    - XPCOM, 444-446
    - XPIInstall, 166-170
    - XUL (XML User Interface Language). *See* XUL
  - msgfmt command, 419
  - MyApp, 216
  - Myers, Scott, 63
  - MyFrame class, 338-339
  - MyThreadFunc() function, 254
- N**
- native GUI toolkits, 325-326. *See also specific toolkits*
  - native IDEs, 67-71
  - .NET Forms, 304, 316, 324
  - Netscape versions, 329
  - Netscape Portable Runtime Library. *See* NSPR
  - networking functions (POSIX), 224
  - NewAESEncryptProxy() function, 513
  - NewClassProxy() function, 507
  - nice() function, 27
  - nmake utility, 81-84
  - nroff command, 192
  - NSHumanReadable Copyright key (Mac OS X), 202
  - NSIS installer
    - application data, 217-218
    - compiling NSIS script, 214
    - creating, 210
    - defining, 210-214
    - distributing, 214
    - documentation, 208
    - documents and settings, 208-209
    - integrating into Start menu and desktop, 216-217
    - program installation, 210
    - testing, 214
    - uninstallers, 215-216
    - Weather Manager sample installer script, 218-220
  - NSPR (Netscape Portable Runtime Library)
    - advantages of, 242-244
    - binary data, 248-249
    - operators, 246-247
    - standard library (libc) functions, 260
    - I/O functions, 261
    - linking functions, 261-263
    - threads, 249
      - advantages of, 250-251
      - creating, 258-260
      - definition of, 250
      - Linux threads, 254-257

- Mac OS X threads, 254-257
  - Win32 threads, 251-254
  - types, 245-246, 296-297
  - when to use, 263-271
  - NtCreateFile() function, 232
  - ntohs() function, 289
  - numbers, floating-point
    - numbers
      - equality, 277-278
      - serializing as binary, 276-277
      - support for IEEE-754 standard, 274-275
  - numeric\_limits template, 295
- O**
- O\_APPEND flag (open() function), 233
  - O\_CREAT flag (open() function), 233
  - O\_EXCL flag (open() function), 233
  - O\_EXLOCK flag (open() function), 233
  - O\_NONBLOCK flag (open() function), 233
  - O\_RDONLY flag (open() function), 233
  - O\_RDWR flag (open() function), 233
  - O\_SHLOCK flag (open() function), 233
  - O\_TRUNC flag (open() function), 233
  - O\_WRONLY flag (open() function), 233
  - objects
    - App, 466-469
    - Document, 472-473
    - DOM (Document Object Model), 434-437
      - interacting with Trixul widgets from JavaScript, 488-493
    - mapping DOM objects to JavaScript, 493-496
    - XUL, 443-444
    - Element, 473-475
  - Observer (publish/subscribe) design pattern, 318-322
  - observers, 319
  - OnApply() function, 401
  - OnBlue() function, 401
  - OnCancel() function, 397
  - OnClickHandler() function, 487
  - OnClickMe() function, 355
  - OnClose() function, 361-362
  - OnColor() function, 409
  - OnDisableEnable() function, 359
  - OnFile() function, 408
  - OnFont() function, 405
  - OnGreen() function, 401
  - OnInit() function, 337
  - OnNewProcess() function, 319-321
  - OnOK() function, 397, 474
  - OnPaint() function, 373-378
  - OnPassword() function, 396
  - OnQuit() function, 355, 359
  - OnRed() function, 401
  - OnSliderChange() function, 388
  - Open Look, 323
  - open command, 185
  - Open Group Base Specifications Issue 6, IEEE Std 1003.1, 2003 Edition, 224-225
  - open() function, 233
  - opendir() function, 236
  - OpenLook, 331
  - OpenPicker class, 447
  - operating systems.
    - See platforms
  - operators (NSPR), 246-247
  - ordering structs, 299-301
  - organizing projects
    - CVS, 45-49, 149-157
    - running, 157
    - setting up, 152-156
    - SVN, 45-49, 149-151
  - OSXCompatibility.cpp file, 106
  - otool utility, 203
  - overriding defaults with site.def (Imake), 99-101
- P**
- p argumen (path program), 161-162
  - parent argument (wxFrame), 365
  - parity, 17-21
  - PassCDebugFlags macro, 130
  - PasswordEntryDialog class, 395-398
  - patch program, 157
    - cross-platform development, 163-164
    - example, 158-160
    - options, 161-162
    - reject files, 162-163
  - pedantic-errors compiler warning (GNU), 63
  - pendantic argument (gcc), 227
  - per-platform compatibility files, 105-106
  - performInstall() function, 169
  - permissions (Linux), 171-173
  - PFloat, 247
  - PkgInfo file, 198

- PL\_CreateOptState()
  - function, 261
- PL\_GetNextOpt()
  - function, 261
- PL\_strdup() function, 261
- PL\_strlen() function, 261
- PL\_strncasestr()
  - function, 261
- PL\_strncpy() function, 261
- PL\_strrstr() function, 261
- platform installs
  - Linux, 170-173
    - /etc/group file, 171
    - /etc/passwd file, 171
  - executing as root, 176-177
  - execution environment, 173-177
  - file permissions, 171-173
  - integrating with
    - GNOME and KDE desktops, 177-186
  - man pages, 186-196
- Mac OS X
  - application layout, 196-202
  - bundle icon files, creating, 207-208
  - drag-and-drop installation, 208
  - MacOS folder, 202-206
- Windows XP NSIS installer
  - application data, 217-218
  - compiling NSIS script, 214
  - creating, 210
  - defining, 210-214
  - distributing, 214
  - documentation, 208
  - documents and settings, 208-209
  - integrating into
    - Start menu and desktop, 216-217
  - program installation, 210
  - testing, 214
  - uninstallers, 215-216
  - Weather Manager sample installer script, 218-220
- platform-specific bugs, tracking, 133
- platforms. *See also*
  - platform installs
    - building source code on multiple platforms, 56-60
    - effect on software portability, 6-8
    - parity, 17-21
    - platform-specific bugs, tracking, 133
    - prioritizing, 17-21
    - sharing code across all supported platforms
      - abstraction layers, 22
      - advantages, 22
      - implementation classes, 31-32
      - instance hierarchies, 42-45
    - Makefiles and building code, 49-52
    - platform factory implementations, 29-31
    - platform-specific ProcessesImpl classes, 32-42
    - ProcessList class example, 23-29
  - testing builds on all supported platforms, 60-61
  - tier-1 platforms, 18, 427
- Platforms screen (Bugzilla), 137
- PO files
  - generating, 417-419
  - locating, 420-421
- pointers, resolving, 507
- policy
  - building source code on multiple platforms, 56-60
  - building source code with multiple compilers, 52-56
  - organizing projects in CVS or SVN, 45-49, 149-151
  - overview, 17
  - parity, 17-21
  - responding to compiler warnings, 61-62
    - GNU flags, 62-63
    - Microsoft Visual C++ flags, 63-64
  - sharing code across all supported platforms
    - abstraction layers, 22
    - advantages, 22
    - implementation classes, 31-32
    - instance hierarchies, 42-45
  - Makefiles and building code, 49-52
  - platform factory implementations, 29-31
  - platform-specific ProcessesImpl classes, 32-42
  - ProcessList class example, 23-29
- testing builds on all supported platforms, 60-61

- portability, 273
  - binary data, 5, 280
    - binary socakddr\_in file, generating, 283-293
  - endian issues, 281
  - intrinsic types and enums, 281
  - struct layout, 281
  - type definitions, 281-282
- build systems, 9
- char types, 278-280
- compilers, 3-4
- configuration
  - management, 9-10
- definition, 1-3
- floating-point numbers
  - equality, 277-278
  - serializing as binary, 276-277
  - support for IEEE-754 standard, 274-275
- language, 3
- operating system
  - interfaces, 6-8
- standard libraries, 5-6
- type sizes
  - efficiency, 297-298
  - integer conversions, 298-299
  - integer types, 293-296
- Netscape Portable Runtime Library (NSPR), 296-297
  - struct alignment and ordering, 299-301
- user interfaces, 8-9
- Portable Operating System Interface for Computer Environments *See* POSIX
- PortableOpen() function, 14
- pos argument
  - (wxFrame), 365
- POSIX, 222-223
  - GNU C library
    - support, 226
  - Microsoft Runtime Library support for, 231-233
  - The Single UNIX Specification, 224-225
  - System V Interface Description (SVID), 223
    - base OS service routines, 223
    - general library functions, 224
    - GNU C library support, 226
    - mathematical functions, 223
    - networking functions, 224
    - string and character handling, 224
- \_POSIX\_C\_SOURCE
  - macro, 230
- \_POSIX\_SOURCE
  - macro, 228-230
- PR\_CreateThread() function, 258-259
- PR\_htonl() function, 249
- PR\_htons() function, 249
- PR\_INT16\_MAX
  - macro, 245
- PR\_INT16\_MIN macro, 245
- PR\_LoadLibrary()
  - function, 261-263
- PR\_ntohl() function, 249
- PR\_ntohs() function, 249
- PR\_OpenDir() function, 269
- PR\_ReadDir() function, 269
- PRDir, 264
- PRDirEntry, 264
- PRDirFlags, 264
- PRFileDesc, 266
- PRInt16, 245
- PRInt16 type, 297
- PRInt32, 245, 266
- PRInt64, 245
- PRInt8, 245
- PrintProcessNameAndID()
  - function, 33
- printsettings-service
  - class, 445
- prioritizing platforms, 17-21
- PRLibrary, 261
- processes, creating, 6-7
- ProcessesFactory class, 28-29, 306-307
- ProcessesImpl class
  - base class, 31
  - CocoaProcessesImpl class, 38-42
  - LinuxProcessesImpl class, 35-38
  - WindowsProcessesImpl class, 32-35
- ProcessList class, 42-45
- ProcessListener class, 319
- Products screen
  - (Bugzilla), 136
- ProecessList class, 23-29
- programs. *See specific programs*
- progressmeter element
  - (XML), 442
- projects, organizing in CVS or SVN, 45-49, 149-151
- property element, 499
- PRStatus, 266-270
- PRThread, 258
- PRThreadPriority, 258
- PRThreadScope, 258
- PRThreadState, 258
- PRThreadType, 258
- PRUint8, 245
- PRUint16, 245, 297
- PRUint32, 245
- PRUint64, 245
- pthread\_attr\_init()
  - function, 257

- pthread\_attr\_setstacksize()  
   function, 257  
 pthread\_create()  
   function, 254-256  
 pthread\_exit()  
   function, 256-257  
 pthread\_join()  
   function, 256-257  
 pthreads, 254  
 publish/subscribe design  
   pattern, 318-322  
 PutBar() function, 85  
 PutBar() function, 78, 100  
 putenv function, 102-105  
 putenv() function, 84
- Q**
- Qt, 324, 331-332  
 QueryInterface()  
   function, 446  
 quit() function, 487
- R**
- r (read) permissions, 172  
 radio element (XML), 441  
 RadioButton class, 448  
 Raymond, Eric, 157  
 read (r) permissions, 172  
 Refresh() function, 401  
 regcomp() function, 269  
 regex function, 265-268  
 regexec() function, 269-270  
 registerChrome()  
   function, 169  
 registering with subjects, 319  
 RegisterListener()  
   function, 320  
 reject files (patch  
   program), 162-163  
 relational operators  
   (NSPR), 246  
 releaseObject() function, 501
- RemoveAESEncryptClass()  
   function, 512  
 rename() function, 241  
 repositories  
   CVS (Concurrent Version  
     System), 149-157  
     organizing projects  
       in, 45-49  
     running, 157  
     setting up, 152-156  
   SVN (Subversion), 45-49,  
     149-151  
 resolving pointers, 507  
 Resources directory (Mac  
   OS X), 204-205  
 responding to user input,  
   388-391  
   EVT\_CHECKBOX  
     macros, 389-390  
   EVT\_COMMAND\_SCR  
     OLL macro, 388  
   GetRadiusPercent()  
     function, 388  
   IsChecked() function, 390  
   OnSliderChange()  
     function, 388  
   SetRadiusPercent()  
     function, 388  
 restricting applications  
   for use with GNOME or  
   KDE, 181  
 return element, 499  
 ReverseValue() function,  
   443-444  
 root, executing Linux  
   installer as, 176-177  
 RTL (Microsoft), 240
- S**
- SaveAsPicker class, 448  
 Scan() function, 23, 26-27,  
   31-33, 39, 43-45, 313  
 ScanProcesses()  
   function, 33-34
- SCM (software configuration  
   management), 131-132  
   bug reporting and  
   tracking systems  
     accessibility, 133  
     Bugzilla, 133-140  
     platform-specific bugs,  
       tracking, 133  
   CVS (Concurrent Version  
     System), 149-157  
     running, 157  
     setting up, 152-156  
   effect on software  
   portability, 9-10  
   importance of, 147-149  
   patch program, 157  
     cross-platform  
       development,  
       163-164  
     example, 158-160  
     options, 161-162  
     reject files, 162-163  
   SVN (Subversion),  
     149-151  
   Tinderbox, 140-147  
     brief log output, 143  
     capabilities, 140,  
       145-147  
     L1 link output, 143  
     lists of check-ins,  
       144-145  
     versions, 140  
   script element (HTML), 433  
   scripting language  
     (DHTML), 433-434  
   scripts  
     build.sh, 78-80  
     install.js, 167-169  
     /usr/bin/mozilla, 174  
   scrolled windows (Trixul),  
     484-485  
   ScrolledView class, 448  
   scrolledview element,  
     484-485  
   ScrolledWindow class, 448

- scrolledwindow element, 484-485
- searching for bugs (Bugzilla), 138-140
- sed command, 75
- SendNotification() function, 320-321
- separate source files
  - for cross-platform development, 85-87
- serialization of binary data, 280
  - binary socakddr\_in file, generating, 283-293
  - endian issues, 281
  - floating-point numbers, 276-277
  - intrinsic types and enums, 281
  - struct layout, 281
  - type definitions, 281-282
- SetBar() function, 78
- SetBrush() function, 374
- setenv() function, 84
- SetGeometry() function, 449
- SetKey() function, 511, 515
- SetLabel() function, 451-453
- SetMenuBar() function, 354, 369
- SetPriority() function, 27
- SetPriorityClass() function, 27
- SetRadiusPercent() function, 388
- SetSizer() function, 355, 385
- SetTopWindow() function, 367
- Set\_key() function, 511
- .SH macro, 192
- SharedLibraryTarget macro, 126
- sharing code across all supported platforms, 22
  - abstraction layers, 22
  - advantages, 22
  - implementation classes, 31-32
  - instance hierarchies, 42-45
  - Makefiles and building code, 49-52
  - platform factory implementations, 29-31
  - platform-specific ProcessesImpl classes, 32-42
  - ProcessList class example, 23-29
- Show() function, 367, 402-403, 449, 453
- ShowModal() function, 395-398, 407
- Shutdown() function, 312, 467-468
- signed char types, 278-280
- SIL files, 509-515
- sin\_family field, 287
- sin\_len field, 287
- sin\_port field, 288
- The Single UNIX Specification, 224-225
- Site.def file, 108
- size argument (wxFrame), 365
- sizeof() function, 294
- sizers (wxWidgets)
  - associating, 385
  - DrawingArea class, 379-384
  - wxBoxSizer class, 378, 384-388
  - wxGridSizer class, 378
  - wxNotebookSizer class, 378
  - wxSlider class, 387
  - wxStaticBoxSizer class, 378
- sizes (type)
  - efficiency, 297-298
  - integer types, 293-296
  - Netscape Portable Runtime Library (NSPR), 296-297
- Skip() function, 362
- skipping events, 359-363
- Smart, Julian, 331
- \$SMPROGRAMS variable, 216
- snprintf() function, 276
- socakddr\_in file, generating, 283-293
  - command-line arguments, 283
  - compiler options, 292-293
  - cross-platform solutions, 288-293
  - DumpAddr() function, 289
  - htons() function, 288
  - ntohs() function, 289
  - on Linux, 287
  - on Mac OS X, 287-288
  - sin\_family field, 287
  - sin\_len field, 287
  - sin\_port field, 288
  - sockaddr\_in struct, 290
  - source code listing, 283-287
  - XPBinaryRead() function, 290-292
  - XPBinaryWrite() function, 290-292
- sockaddr\_in struct, 290
- software configuration management. *See* SCM
- software installation, 165-166
  - Linux platform installs, 170-173
  - /etc/group file, 171
  - /etc/password file, 171

- executing as root, 176-177
- execution
  - environment, 173-177
- file permissions, 171-173
- integrating with
  - GNOME and KDE desktops, 177-186
  - man pages, 186-196
- Mac OS X platform
  - installs
    - application layout, 196-202
    - bundle icon files, creating, 207-208
    - drag-and-drop installation, 208
    - MacOS folder, 202-206
  - Windows XP NSIS installer
    - application data, 217-218
    - compiling NSIS script, 214
    - creating, 210
    - defining, 210-214
    - distributing, 214
    - documentation, 208
    - documents and settings, 208-209
    - integrating into
      - Start menu and desktop, 216-217
    - program
      - installation, 210
    - testing, 214
    - uninstallers, 215-216
    - Weather Manager
      - sample installer script, 218-220
    - XPIInstall, 166-170
- SomeFuncInt() function, 279**
- source files for
  - cross-platform development, 85-87
- source trees, building with
  - Imake, 108-111**
    - compatibility library
      - Imakefiles, 116-119
    - Makefiles, 119-130
    - Windows
      - implementations, 111-116
- SourceVersion key (Mac OS X), 198**
- Spacer class, 448**
- spacer element, 440, 480**
- SpiderMonkey JavaScript engine, 494**
- standard libraries
  - effect on software
    - portability, 5-6
  - STL (Standard Template Library), 5-6
- standards-based APIs
  - BSD (Berkeley Standard Distribution), 225
  - choosing, 240-241
  - GCC standards support
    - compiler flags, 227
    - Cygwin, 238-240
    - headers, 227-228
    - macros, 228-231
    - MinGW, 234-237
  - GNU C library, 226
  - POSIX, 222-223
    - Microsoft Runtime Library support
      - for, 231-233
    - System V Interface Description (SVID), 223-224
  - The Single UNIX Specification, 224-225
- XPG (X/Open Portability Guide), 225
- Start menu, adding applications to, 178-181
- StaticText class, 448**
- std=c++98 compiler warning (GNU), 63
- STL (Standard Template Library), 5-6
- strftime() function, 88-89**
- string functions (POSIX), 224
- Stroustrup, Bjarne, 299
- strtod() function, 276**
- structs
  - alignment and ordering, 299-301
  - binary data issues, 281
  - JClass, 507
  - numeric\_limits, 295
  - sockaddr\_in, 290
  - xpsockaddr\_in, 290
- style argument (wxFrame), 365
- styles, CSS (Cascading Style Sheets), 437-438
- subdirectories (project), building with Imake, 108-111
  - compatibility library
    - Imakefiles, 116-119
  - Makefiles, 119-130
  - Windows
    - implementations, 111-116
- subjects, 319
- subscribe. *See* publish/subscribe design pattern
- Subversion (SVN), 10, 45-49, 149
- Summerfield, Mark, 331
- SunOS Open Look, 323
- Sutter, Herb, 276

- SVID (System V Interface Description)**  
 base OS service routines, 223  
 general library functions, 224  
 GNU C library support, 226  
 mathematical functions, 223  
 networking functions, 224  
 string and character handling, 224  
**\_SVID\_SOURCE** macro, 230  
**SVN (Subversion)**, 10, 45-49, 149  
**sysctl()** function, 38  
 system calls, effect on software portability, 6-8  
**System V Interface Description**. *See* **SVID**
- T**
- templates**  
 numeric\_limits, 295  
 STL (Standard Template Library), 5-6  
**Terminal**, 196-197  
**testing**  
 building source code with multiple compilers, 53  
 Cygwin, 239-240  
 MinGW, 235-236  
 NSIS installer, 214  
 on all supported platforms, 60-61  
**Text class**, 448  
**textbox** element (XML), 440  
**.TH** macro, 192  
**themes (GUIs)**, 304-305  
**threads**, 249  
 advantages of, 250-251  
 definition of, 250  
 Linux threads, 254-257  
 Mac OS X threads, 254-257  
 NSPR threads, 257-260  
 Win32 threads, 251-254  
**tier-1 platforms**, 18, 427  
**Tinderbox**, 21, 140-147  
 brief log output, 143  
 capabilities, 140, 145-147  
 L1 link output, 143  
 lists of check-ins, 144-145  
 versions, 140  
**title argument (wxFrame)**, 365  
**title element (HTML)**, 430  
**toolbar** element (XML), 440  
**toolbarbutton** element (XML), 440  
**toolbars (XUL)**, 440  
**toolchains**  
 autoconf/automake, 87-90  
 compilers, 66-67  
 Cygwin, 71-76  
 definition of, 65-66  
**Imake**  
 bar program example, 95-98  
 building debug, 130  
 building projects with subdirectories, 108-130  
 darwin.cf file, 107  
 darwin.rules file, 107  
 eliminating #ifdefs in code, 101-106  
 Hello World example, 93-94  
 Imake.rules file, 107  
 Imake.tpl file, 107  
 Imakefiles, 94-95, 108  
 installing on Mac OS X, 91  
 installing on Windows, 91-93  
 linux.cf file, 107  
 Makefiles, 108  
 overriding defaults with site.def, 99-101  
 Site.def file, 108  
 Win32.cf file, 107  
 Win32.rules file, 107  
**make** utility  
 bar program example, 77-78  
 compared to build.sh script, 78-80  
 Makefiles, 79-81  
 native IDEs, 67-71  
 nmake utility, 81-84  
**toolkits (GUIs)**, 324-325.  
*See also specific toolkits*  
 cross-platform GUI toolkits, 326  
 custom GUI toolkits, 327  
 native GUI toolkits, 325-326  
**TopLevel** class, 364  
**Trixul**, 327  
 application main loop, 466-469  
 App object, 466-469  
 AppImpl class, 467  
 in Cocoa, 469  
 in Gtk+, 470  
 in Windows, 470-471  
 basic operation, 448-449  
 capabilities, 447  
 documents, 472-473  
 elements, 473-475  
 goals, 446-447  
 implementation classes, 452  
 ButtonImpl, 453-456  
 CocoaButtonImpl, 456-457  
 GtkButtonImpl, 457-458  
 WidgetImpl, 454-455  
 WindowsButtonImpl, 458-459



- implementation
    - objects, 459
    - in Cocoa, 460-461
    - in Gtk+, 461
    - in Windows, 462-463
  - integration with C++
    - components, 496-497
      - accessing components from JavaScript, 500-502
    - coding C++
      - components, 502-507
    - creating base
      - JavaScript objects, 507-508
    - defining JavaScript
      - functions, 508-509
    - describing components
      - in XML, 498-500
    - generating source
      - code from SIL files, 509-515
    - instantiating C++
      - objects, 508
    - linking and
      - distributing
        - components, 517
    - resolving points to
      - creator/destroyer
        - functions, 507
    - step-by-step sequence, 497-498
    - writing C++ code
      - to implement
        - components, 515-517
  - integration with
    - JavaScript, 485-486
      - example, 486-487
      - including external
        - JavaScript sources, 487-488
    - interacting with
      - widgets from
        - JavaScript, 488-493
      - mapping DOM objects
        - to JavaScript, 493-496
    - layouts, 477, 480-483
    - scrolled windows, 484-485
    - UI creation, 471-472
    - widget creation, 475-477
    - widget definitions, 449-452
    - widget factories, 463-466
      - CocoaFactory class, 465-466
      - GetWidgetFactory()
        - function, 464
      - WidgetFactory class, 463-465
    - widget support, 447-448
  - Trolltech, 324**
  - troubleshooting**
    - bug reporting and
      - tracking systems, 133
    - accessibility, 133
    - Bugzilla, 133-140
    - platform-specific bugs,
      - tracking, 133
    - compiler warnings, 61-62
      - GNU flags, 62-63
      - Microsoft Visual C++
        - flags, 63-64
  - types**
    - DIR, 229-230
    - NSPR types, 245-246
    - sizes, 293-298
  - U**
  - UIs (user interfaces). See GUIs (graphical user interfaces)**
  - ul element (HTML), 431**
  - Umbrello, 42**
  - uname command, 50**
  - uninstallers (NSIS), 215-216**
  - UninstPage command, 216**
  - unregistering from subjects, 319**
  - UnregisterListener() function, 320**
  - unsigned char types, 278-280**
  - UppercaseValue() function, 443-444**
  - user interfaces. See GUIs (graphical user interfaces)**
  - /usr/bin/mozilla script, 174**
  - /usr/local/man directory, 195**
  - utilities. See command-line utilities**
- V**
- variables**
    - \$APPDATA, 218
    - \$SMPROGRAMS, 216
    - m\_enabled, 358
  - vbox element (XML), 439-440**
  - verifying MinGW libraries, 236-237**
  - version.plist file, 197-198**
  - view command, 185**
  - views (GUIs), 306**
    - abstracting GUI main
      - loop, 309-313
    - CreateUI()
      - function, 312
    - gtk\_init()
      - function, 310
    - gtk\_main()
      - function, 310
    - GUI class, 310-313
    - Init() function, 312
    - MainLoop()
      - function, 312
    - m\_guiImpl
      - member, 311
    - Shutdown()
      - function, 312

- CocoaGUIImpl class, 315
- CreateMenus()
  - function, 317-318
- GUIImpl class, 314
- illustration of UI
  - implementation, 306
- LinuxFactory class, 307-308
- LinuxGUIImpl class, 308, 317
- ProcessesFactory
  - class, 306-307
- WindowsGUIImpl
  - class, 316
- Visual C++**
  - compiler warnings, 63-64
  - Imake for Visual C++, installing, 92-93
- W**
- w (write) permissions, 172
- Wall argument (gcc), 227
- Wall compiler warning (GNU), 63
- /Wall flag (Visual C++), 64
- warnings (compilers), 61-62
  - GNU flags, 62-63
  - Microsoft Visual C++ flags, 63-64
- Weather Manager sample
  - installer script, 218-220
- Werror flag, 63
- Werror compiler warning (GNU), 63
- WidgetFactory class, 463-465
- WidgetImpl class, 454-455
- widgets, Trixul
  - Button, 449-452
  - container widgets, 450
  - control widgets, 450
  - creating, 475-477
  - defining, 449-452
  - factories, 463-466
    - CocoaFactory class, 465-466
    - GetWidgetFactory()
      - function, 464
    - WidgetFactory class, 463-465
  - implementation
    - classes, 452
    - ButtonImpl, 453-456
    - CocoaButtonImpl, 456-457
    - GtkButtonImpl, 457-458
    - WidgetImpl, 454-455
    - WindowsButtonImpl, 458-459
  - implementation
    - objects, 459
    - in Cocoa, 460-461
    - in Gtk+, 461
    - in Windows, 462-463
  - layouts, 477, 480-483
  - scrolled windows, 484-485
  - support for, 447-448
  - XUL, 442
- widgets, wxWidgets. *See* wxWidgets
- Win32, 251-254, 304, 324
- Win32.cf file, 107
- Win32.rules file, 107
- Window class, 448
- window element (XML), 439
- Windows**
  - building source code on
    - conditional
      - compilation, 84-85
    - nmake utility, 81-84
    - separate source files, 85-87
  - building wxWidget
    - applications on, 348-349
- Cygwin, 71-76, 238
  - downloading and installing, 238-239
  - testing, 239-240
- Imake installation
  - Imake for GCC, 91-92
  - Imake for Visual C++, 92-93
- MFC (Microsoft Foundation Classes), 324
- MinGW
  - definition of, 234
  - downloading and installing, 234
  - testing, 235-236
  - verifying libraries, 236-237
- .NET Forms, 324
- project source trees, building, 111-116
- Trixul application main
  - loops, 470-471
- Trixul implementation
  - objects, 462-463
- Win32, 251-254, 304, 324
- Windows XP NSIS
  - installer
    - application data, 217-218
    - compiling NSIS script, 214
    - creating, 210
    - defining, 210-214
    - distributing, 214
    - documentation, 208
    - documents and settings, 208-209
    - integrating into Start menu and desktop, 216-217
    - program
      - installation, 210

- testing, 214
- uninstallers, 215-216
- Weather Manager
  - sample installer script, 218-220
- WindowsButtonImpl class, 458-459
- WindowsFactory class, 29-31
- WindowsGUIImpl
  - GUI implementation class, 316
- WindowsProcessesImpl class, 32-35
- wxWidget installation, 334-335
- windows (XUL), 439
- Windows XP NSIS installer
  - application data, 217-218
  - compiling NSIS script, 214
  - creating, 210
  - defining, 210-214
  - distributing, 214
  - documentation, 208
  - documents and settings, 208-209
  - integrating into Start menu and desktop, 216-217
  - program installation, 210
  - testing, 214
  - uninstallers, 215-216
  - Weather Manager sample installer script, 218-220
- WindowsButtonImpl class, 458-459
- WindowsFactory class, 29-31
- WindowsGUIImpl class, 316
- WindowsProcessesImpl class, 32-35
- /Wn flag (Visual C++), 64
- write (w) permissions, 172
- wx-config command, 333
- /WX flag (Visual C++), 64
- wxALIGN\_CENTER argument (Add() function), 343
- wxAND\_INVERT value (logicalFun argument), 376
- wxAND\_REVERSE value (logicalFun argument), 376
- wxAND value (logicalFun argument), 376
- wxApp class, 337-338
- wxBoxSizer class, 336, 341-345, 378, 384-388
- wxBrush class, 374
- wxCAPTION argument (wxFrame), 365
- wxCHANGE\_DIR style (wxFileDialog), 406
- wxCLEAR value (logicalFun argument), 376
- wxCLIP\_CHILDREN argument (wxFrame), 365
- wxColour class, 374
- wxColourDialog class, 409-410
- wxCommandEvent events, 351
- wxCOPY value (logicalFun argument), 376
- wxEUIV value (logicalFun argument), 376
- wxEvtHandler class, 336
- wxFileDialog class, 406-408
- wxFontDialog class, 404-405
- wxFrame class, 336, 339, 364-366
  - constructor, 364-365
  - sample implementation, 364-367
- wxFRAME\_FLOAT\_ON\_PARENT argument (wxFrame), 366
- wxFRAME\_NO\_TASKBAR argument (wxFrame), 366
- wxFRAME\_SHAPED argument (wxFrame), 366
- wxFRAME\_TOOL\_WINDOW argument (wxFrame), 366
- wxGetTranslation() function, 417
- wxGridSizer class, 378
- wxHIDE\_READONLY style (wxFileDialog), 406
- wxICONIZE argument (wxFrame), 365
- wxINVERT value (logicalFun argument), 376
- wxLanguage enums, 422-424
- wxLANGUAGE\_\* values (wxLanguage enum), 422-424
- wxLocale class, 421-422
- wxMAXIMIZE argument (wxFrame), 365
- wxMAXIMIZE\_BOX argument (wxFrame), 365
- wxMemoryDC class, 375
- wxMenu class, 368-369
- wxMenuBar class, 369-370
- wxMessageBox class, 393-394
- wxMINIMIZE argument (wxFrame), 365
- wxMINIMIZE\_BOX argument (wxFrame), 365
- wxMULTIPLE style (wxFileDialog), 406
- wxNAND value (logicalFun argument), 376
- wxNOR value (logicalFun argument), 376
- wxNotebookSizer class, 378
- wxNO\_OP value (logicalFun argument), 377
- wxOPEN style (wxFileDialog), 406

- wxOR\_INVERT value (logicalFun argument), 377
- wxOR\_REVERSE value (logicalFun argument), 377
- wxOR value (logicalFun argument), 377
- wxOVERWRITE\_PROMPT style (wxFileDialog), 406
- wxPaint class, 373
- wxPaintDC class, 373-375
- wxRESIZE\_BORDER argument (wxFrame), 366
- wxSAVE style (wxFileDialog), 406
- wxSET value (logicalFun argument), 377
- wxSlider class, 387
- wxSRC\_INVERT value (logicalFun argument), 377
- wxStaticBoxSizer class, 378
- wxStaticText class, 336, 340-341
- wxSTAY\_ON\_TOP argument (wxFrame), 365
- wxSYSTEM\_MENU argument (wxFrame), 365
- wxTextEntryDialog class, 395
- wxWidgets, 329-331
  - building applications, 345
    - on Linux, 347-348
    - on Mac OS X, 348
    - on Windows, 348-349
  - container widgets, 363-364
  - definition of, 331
  - dialogs
    - ColorDialog class, 400-404
    - modal dialogs, 392-398
    - modeless dialogs, 399-404
    - PasswordEntryDialog class, 395-398
  - wxTextEntryDialog class, 395
  - wxTextEntryDialog class, 395
- downloading, 331
- DrawingArea class, 379-383, 391
- events, 349
  - BEGIN\_EVENT\_TAB LE macro, 352-353
  - Click Me button, creating, 354-355
  - DECLARE\_EVENT\_TABLE macro, 350-351
  - Enable/Disable button, creating, 356-359
  - END\_EVENT\_TABLE macro, 352-353
  - event handler function implementations, 355
  - EVT\_BUTTON macro, 353
  - EVT\_MENU macro, 353
  - File menu, 354
  - skipping, 359-363
  - wxCommandEvent events, 351
- frames
  - drawing content in, 373-378
  - wxFrame class, 364-367
- Hello World program, 335
  - Add() function, 342-344
  - building on Linux, 347-348
  - building on Mac OS X, 348
  - building on Windows, 348-349
  - container widgets, 336-337
  - control widgets, 336
  - main window, 338-345
  - MyFrame class, 338-339
  - OnInit() function, 338
  - source code listing, 345-346
  - wxBoxSizer class, 341-345
  - wxFrame class, 339
  - wxStaticText class, 340-341
- history and development, 331
- IMPLEMENT\_APP macro, 338
- installing, 332
  - on Linux, 334
  - on Mac OS X, 333
  - on Windows, 334-335
- internationalization, 410-411
  - gettext, 411-420
  - locales, creating, 421-425
  - message files, loading at runtime, 425-426
  - PO/MO files, 417-421
  - wxLanguage enums, 422-424
  - wxLocale class, 421-422
- licensing terms, 331-332
- menus, 367
  - associating with functions, 370-372
  - creating with wxMenu class, 368-369
  - menu bars, 369-370
- online documentation, 331
- responding to user input, 388-391
  - EVT\_CHECKBOX macros, 389-390

- EVT\_COMMAND\_SCROLL
    - macro, 388
  - GetRadiusPercent()
    - function, 388
  - IsChecked()
    - function, 390
  - OnSliderChange()
    - function, 388
  - SetRadiusPercent()
    - function, 388
  - wxApp class, 337-338
  - wxBoxSizer class, 336, 341-345, 378, 384-388
  - wxBrush class, 374
  - wxColour class, 374
  - wxColourDialog class, 409-410
  - wxEvtHandler class, 336
  - wxFileDialog class, 406-408
  - wxFontDialog class, 404-405
  - wxFrame class, 336, 339, 364-366
    - constructor, 364-365
    - sample implementation, 364-367
  - wxGridSizer class, 378
  - wxLocale class, 421-422
  - wxMemoryDC class, 375
  - wxMenu class, 368-369
  - wxMenuBar class, 369-370
  - wxMessageBox class, 393-394
  - wxNotebookSizer class, 378
  - wxPaint class, 373
  - wxPaintDC class, 373-375
  - wxSlider class, 387
  - wxStaticBoxSizer class, 378
  - wxStaticText class, 336, 340-341
  - wxWindow class, 336
  - wxXOR value (logicalFun argument), 377
- X-Y-Z**
- x (execute) permissions, 172
  - X/Open Portability Guide (XPG), 224-226
  - X11, 324
  - X11R6, 111
  - XBD, 225
  - Xcode IDE, 68
  - xmkmf command, 96-98
  - XML elements. *See* elements
  - XML User Interface Language. *See* XUL
  - \_XOPEN\_SOURCE
    - macro, 230
  - XP\_PUTENV, 102
  - XP\_WIN, 85
  - XPBinaryRead() function, 290-292
  - XPBinaryWrite() function, 290-292
  - XPCOM, 444-446
  - XPConnect, 444-446
  - XPG (X/Open Portability Guide), 224-226
  - XPI files, 167
  - XPInstall, 166-170
  - xpsockaddr\_in struct, 290
  - XSH, 225
  - XUL (XML User Interface Language), 428-429
    - boxes, 439-441
    - controls, 441
    - DHTML (Dynamic HTML)
      - CSS (Cascading Style Sheets), 437-438
      - DOM (Document Object Model), 434-437
    - HTML (Hypertext Markup Language), 429-432
      - scripting language, 433-434
    - dialogs, 439
    - DOM (Document Object Model), 443-444
    - JavaScript, 443-444
    - menus, 441-442
    - programming with, 442
    - toolbars, 440
    - toolkit, 327
    - widgets, 442
    - windows, 439
    - XPCOM, 444-446
    - XPConnect, 444-446
  - XView, 331
  - /Za flag (Visual C++), 64