

Index

2G networks, 16, 30
 2.5G networks, 16, 30
 3D mobile games, 12
 3G networks, 16
 3gpp format, 347

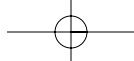
A

Active state, for MIDlets, 61
 Advanced Graphics Optional Package, 48
 Alert class, 125–28
 Alert constructors, 125
 AlertTypes, predefined, 125
 Alpha blending, 235
 AM_ADDR, 374, 377
 AMS (application management software), 49,
 61, 63–64, 96–97, 312
 Anchors, listed, 147
 Animated transit screens, 284–87
 AnimatedWaitViewer package, 284
 Animation, 154–56, 157f, 195, 207–09, 216–17
 Antenna
 Ant tasks supported, 491–92t
 build3.xml file, full listing of, 502–06
 bytecode obfuscation, 500–502
 directive usage, examples of, 494–97
 installing, 490–91

postprocessing, 499–500
 preprocessing and postprocessing, need
 for, 488–89
 preprocessor directives, 492, 493t
 smart linking, 499–500
 third-party software requirements, 490
 wtkobfuscate task, 500–501
 wtkpreprocess task, 492, 497–99
 wtksmartlink task, 499–500

Apache Ant

automated build example, 93–96
 build script, running, 95
 build targets, 94–95
 core tasks, 83
 environment variables, configuring, 83–84
 IDE support for, 85, 86
 IDEs, integrating scripts into, 95–96
 installing, 83
 JAR size and JAD attribute, matching, 95
 path directory separators, 94
 API access permission MIDlet attributes, 103
 AppForge, 15
 ARGB arrays, image conversion to/from,
 165–66
 ARGB image data format, 164
 assert methods, 520
 Audio playback, MIDI player. *See* MIDI player
 MIDlet
 Authenticated devices, 377
 Authentication MIDlet attributes, 100–101



B

bash, 88

BCC (Bluetooth Control Center), 377, 386, 390

BD_ADDR, 374, 377

Binary FM modulation, 373

BinaryMessage interface, 308–09

Bloggers, 12, 356

Bluetooth Host Controller Interface(HCI),
379–80

Bluetooth SIG (Special Interest Group), 373

Bluetooth technology

application testing, in an emulated
environment, 409–10

authenticated devices, 377

band operation, 373

BCC (Bluetooth Control Center), 377, 386,
390

BTDiscovery application, discussed, 402–09

cached devices, 377

class of device (CoD), 382–84

connectivity confirmations, suppressing,
102

data integrity, 373

data throughput, 373

distance range, 373

Host Controller Interface(HCI), 379–80

inquiry procedure, 382–83

interference, 374

known or preknown devices, 377

links, physical and logical, 373

paired devices, 377

physical channels, 373

power classes and consumption, 373–74

profiles, 373, 381

protocol stack, 379–80

as provisioning option, 107

Series 40 device support, 30

service discovery, 384

slots, 373

standardization, 373

Symbian OS communication layer, 55

as wireless networking option, 16, 18

Bluetooth technology, Java API (JABWT),
390, 397–99

client connection example, 401–02

client connection, setting up, 388

configuration menu and the BCC, 386

connection URL parameters, 390

DataElement attribute IDs, 396

discover services, remote devices, 397–99

discovery mode parameter, 391–92

exceptions, catching, 387

GCF connection interfaces and factory
classes, 387

JSR 82, 384–85

L2CAP, connection types for, 387

local devices, getting information about,
391–93

network devices, discovery process for,
393–94

PREKNOWN and CACHED devices, 394

register services, server connections, 397

remote devices, getting information about,
394–95

runtime properties, querying, 391, 392t

server application example, 399–401

server connection, setting up, 387–88

Service Discovery Database (SDDDB), 395–97

service records, 395–97

stack, initializing with configuration utility,
385

submenu options, 386

URL connection strings, 387

the UUID (universally unique identifier),
387–88, 388–89t

Bluetooth technology, piconets

active member address (AM_ADDR), 374, 377

Bluetooth device address (BD_ADDR), 374,
377

connection modes, 377

master devices and slave nodes, 374–76

scatternets, 375–76

sniff interval, Nokia, 377

Bluetooth technology, security

authentication process, 378–79

device authorization, per-use basis, 379

encryption, 379

pairing process, 378

Boeing, 7

bootclasspath options, 87–88

Borland Enterprise Studio for Mobile, 85

Borland JBuilder, 85

Borland Mobile Studio, 85

Browser applications. *See also* WAP

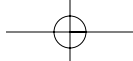
browsers/gateways

authoring content, Nokia devices, 564–65

content download and upload, 567–69

making a phone call, 573

Nokia browser developer tools, 569–70,
571f



- phone book entries, adding, 574
- sending a DTMF tone, 573–74
- skill migration issues, 14
- thin-client application paradigm, 560–61
- wallet, 574–75
- WML versus XHTML MP, 565–67
- WMLScript, 567
- BTDiscovery application, discussed, 402–09
- build3.xml file, 502–06
- Byte arrays, 242, 245
- ByteArrayInputStream class, 245
- ByteArrayOutputStream class, 245
- Bytecode obfuscation, 500–502

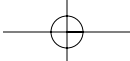
C

- Cached devices, 377, 394
- CaliberRM, 85
- cancel() method, 75, 76
- Canvas class, 145
- Canvas object, 64
- Canvas.hideNotify() method, 182
- Canvas.showNotify() method, 182
- <card> tag, 40, 41f
- CDMA networks, 16, 30
- Certificate interface, 303
- Chain of responsibility pattern, 296
- Chat example application
 - native SMS versus chat MIDlet suite, 315
 - receiving messages, 320–23
 - sending messages, 317–20
 - SMS emulation, 315–17
- checkPermission() method, 64, 102
- Choice interface, static constants, 121
- ChoiceGroup class, 137–39
- Class of device (CoD), 382–84
- classpath option, 87–88
- Clipping, 147, 174, 221, 222
- closeRecordStore() method, 248, 249
- collidesWith() method, 203–04
- Collision detection, 203–04
- colorSpecifier argument, 157–58
- Command buttons, 69
- Command class constructors, 116
- Command object, 64–65, 70, 115–16, 116–17, 158–59
- Command placement, 158–60
- Command types, 117
- commandAction() method, 69, 74, 115–16,

- 154
- Command line
 - Ant build script, running, 95
 - compiling command, 87–88
 - shells, 88
 - tools, 86
 - window, bringing up, 88
- CommandListener interface, 65, 69, 114–16, 423–24
- CommandListener.commandAction()
 - callback, 65
- Command-related methods, 116, 185
- CommConnection interface, 268
- Com.nokia.mid.ui, 184–87
- Compiler errors, 510–11
- Compiling command, 87–88
- Concept SDKs, 81, 515
- Configurations, 47, 48
- Connected Device Configuration (CDC), 48
- Connected Limited Device Configuration (CLDC), 48
- Connection interface, 262–63
- Connector class, 261–63
- Connector.open() static method, 261, 263
- Consumer applications, 6
- Cookies. *See also* HTTP headers
 - client-side tracking, 289–92
 - server-side, 288–89
- Copyright protection, 12
- createAnimatedTile() method, 216–17
- createPlayer() method, 329–33
- CRM (customer relationship management)
 - software, 7
- Crossplatform, defined, 46
- csh, 88
- CSS (cascading style sheets), 41
- CustomItem class, 167–71
- Cyclic Error Check (CRC), 373
- Cygwin open source project, 88

D

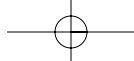
- Data storage. *See also* Photo Viewer example
 - byte arrays, 242, 245
 - persistent storage, 241–42
 - Random Access Memory (RAM), 241
 - read-only storage, JAD and JAR manifest files, 241–42
 - remote storage, 242



- Data storage (*cont.*):
 RMS (Record Management System), 242,
 247. *See also* `RecordStore`
 serialization, Java objects, 242, 244–47
- Data synchronization, 460–61
- Datagrams, 268–69
- `DataInputStream`, 264
- `DataInputStream` object, 245
- `DataOutputStream`, 264
- `DataOutputStream` object, 245
- `DataSource` class, 332
- `DateField` class, 136–37
- Debugging. *See also* J2MEUnit framework;
 UI testing guidelines; Unit testing
 common errors, 510–11
`Exception.printStackTrace()` method,
 514
 Java Debug Wire Protocol (JDWP), 511
 Java Virtual Machine Debugger Interface
 (JVMDI), 511
`MIDPLogger`, 515–19
 print statements, 512–15
`System.out.println()` method, 513–14
- Dedicated Inquiry Access Code (DIAC),
 382–83
- `defineCollisionRectangle()` method,
 204
- delay argument, 76
- `delayUntilNextFrame()` method, 195, 223
- Design patterns
 data synchronization, 460–61
 Model-View-Controller (MVC) pattern,
 423–25
 object creation, 434
 Singleton pattern, 439
 static class pattern, 434–35
- `destroy()` method, 70
- `destroyApp()` method, 62, 70
- Destroyed state, for MIDlets, 61
- Develop-and-optimize process. *See also*
 Debugging
 API availability, 480–81
 enabler technologies, selecting, 477
 the GCF, 482
 Heap memory, 480
 JAR size limit, 479–80
 languages and cultures, 483
 the MMAPAPI, 481–82
 RMS space, 480
 screen characteristics, 478–79
 Series 40/Series 60 scalability, 477–78
 target application, building a generic
 version, 477
 target devices, listing, 478
 thread behavior, 482
 UI component behavior, 482
- Developer skill migration paths
 Java desktop developers, 15
 Java server-side developers, 14–15
 Visual Basic corporate developers, 15
- Developer tools. *See* J2ME development
 tools
- Developing Series 60 Applications: A Guide
 for Symbian OS C++ Developers
 (Edwards and Baker), 15
- Device controls, advanced, 156–58
- Device SDKs. *See* SDKs
- `DeviceControl` utility class, 186
- Digital Rights Management (DRM), 26
- Digital signatures
`Certificate` interface, 303
 and HTTPS, 301–02
 JAD file, 90, 97
 MIDlet suite authentication, 100–101
- Digital voice and analog data networks (2G
 networks), 16
- `DirectUtils` utility class, 186
- `DiscoveryAgent` class, 393
- `DiscoveryListener` class, 393
- Discrete event simulation, 226
- `Display` class, 64, 112–13
- `Displayable` class, 113–16, 423
- `Displayable` object, 64–65, 113, 115, 424
- `Display.getDisplay()` method, 112
- DoCoMo, 12
- Double-buffering, 146, 191, 195–97, 221
- Drawing methods, `Graphics` class, 146–47
- `drawPixels()` method, 187
- `drawPolygon` method, 187
- `drawRGB()` method, 166
- DTMF tones, sending, 573–74
- duration argument, 332
- DVD players, 6

E

- Eclipse, 85–86
- Embedded databases, 422
- Enhanced Data Rates for Global Evolution
 (EDGE) networks, 16, 30



Ericsson, 373
Event strings, 336

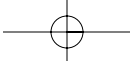
F

FedEx package tracking, 7
File Transfer Profile (FTP), 381
`fillPolygon` method, 187
Fish game example
 animating fish, 212–13
 animation speed, regulating, 223–24
 background, animated `TiledLayer`, 217–20
 collision detection, 205–07
 drawing attached layers to the screen, 221–22
 fish constructor, 210
 layers, attaching to a `LayerManager`, 221
 loading fish image, 212
 manta rays, adding, 230–34
 `move()` method, 201–02
 pictured, 196f
 `readInput()` method implementation, 199
 starfish, adding, 226–29
`flushGraphics()` method, 195–97, 221–22
Fonts, 148
Form class, 130
Form object, 64, 69
Forum Nokia
 device specification documents, 32
 Mobile Media API Technical Note, 331
 NDS, downloading, 78
 SDKs, downloading, 80
 total device storage information, 247
 value chain management, 11
Forum Nokia PRO, 11, 56
Forward Error Correcting (FEC), 373
Foundation Profile, 48
Frames, 202, 203f, 207–09, 224
Fujitsu, 53
`FullCanvas` class, 184–86

G

Game API. *See also* `GameCanvas`; `Sprites`;
 `TiledLayer`
 animation speed, regulating, 223–24
 `LayerManager`, 206, 220–21

 layers, 200–201
 z-order, 220
Game key states, 198
`GameCanvas`
 callbacks, suppressing unnecessary, 199
 full-screen mode, 197
 game actions versus literal keys, 197–98
 `Graphics` buffer, 195–97
 main game processing, 191, 193–95
 simultaneous key presses, 199
Games. *See also* Fish game example; Game
 API; Trivia game example
 action key mapping, 150–51, 197–98
 animation speed, regulating, 223–24
 backlight, 235
 components, 191
 discrete event simulation, 226
 double-buffering, 191, 195–97, 221
 frames, dropping, 224
 the game loop, 190
 `GameCanvas` class, 191
 high frequency sampling, 225
 LCDUI low-level API, 111–12
 menu, placing MIDlet suite under, 105–06
 MIDlet, skeleton code for, 191–92
 multiplayer and 3D visual, 12
 simultaneous key presses, 199
 splash screens, 191, 192
 thread, for main game loop, 193–95
 transparency and pixel arrays, 235
 transparent images, importance of, 164
 usage model, 415
Garbage collection, 45–46, 223, 241, 434
Gauge object, 139–42
`GaugeWaitViewer` package, 279
`gb2312` encoding, 265
General Inquiry Access Code (GIAC), 382, 391
General Packet Radio Service (GPRS), 16
Generic Access Profile (GAP), 381
Generic Connection Framework (GCF) API.
 See also Cookies; HTTP headers;
 `HttpClient`; `HttpConnection`
 interface; Photo Viewer example,
 networked
 Certificate interface, 303
 `CommConnection` interface, 268
 `Connection` interface, 262–63
 connectivity, importance of, 260
 `Connector` class, 261–63
 `Connector.open()` static method, 261, 263
 `HttpsConnection` interface, 266, 301–02



optimizing, 482
 SecureConnection interface, 267
 SecurityInfo interface, 302
 serial ports, retrieving list of, 268
 ServerSocketConnection interface, 267
 SocketConnection interface, 266
 static factory methods, use of, 261
 supported protocols, 261, 263
 UDPDatagramConnection interface, 268–69
 URL schemes, 261–62
 WMA integration, 309
 Generic Object Exchange Profile (GOEP), 381
 Generic visual thread framework. *See also*
 Trivia game example; Worker threads
 code for use of, 451
 components, 450f
 visual threads, examples, 451f
 WaitScreen class, 453–54
 WorkerRunnable interface, 452
 WorkerThread class, 452
 getAppProperty() method, 63, 69
 getCurrent() method, 113
 getFont() method, 148
 getFrameSequenceLength() method, 208
 getKeyStates() method, 197–98
 getPixels() method, 187
 getRawFrameCount() method, 208
 GIF images, 133
 Graphics class, 146–47
 Graphics.copyArea() method, 166
 Graphics.drawRegion() method, 166
 GSM Data networks, 16, 309

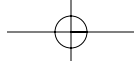
H

Header Error Check (HEC), 373
 High frequency sampling, 225
 Hold mode, 377
 HSQLDB, 421–23
 HTML pages, retrieving, 264–65
 HTTP authentication, 98
 HTTP GET operation, 264
 HTTP headers. *See also* Cookies
 authentication protocol, 295
 handlers, implementing, 298–99
 HttpClient framework, 295–98
 querying, 265–66
 session tracking, 287–88
 HTTP POST operation, 98, 103, 264

HttpClient
 chain of responsibility pattern, 296
 framework, 295–98
 header handlers, implementing, 298–99
 utility, 295–301
 HttpURLConnection interface
 data streams, opening, 264
 GET and POST operations, 264
 HTML pages, retrieving, 264–65
 HTTP headers, 265–66
 metadata, 265
 status codes, 265
 text versus encoded content, querying, 265
 URL scheme, 264
 HTTPSConnection interface, 266, 301–02

I

IBM
 Bluetooth SIG (Special Interest Group), 373
 mobile sales solutions, 7
 WMQe, 307
 ibus//Mobile, 307
 IDEs (Integrated Development Environments)
 Ant scripts, integrating, 95–96
 Apache Ant support, 85, 86
 Borland JBuilder, 85
 Eclipse, 85–86
 NDS integration with, 78
 Sun Java Studio, 85
 visual RAD tool support, 15
 Image transforms, 209–10
 Image transparency, 164, 165f
 ImageItem class, 131–33, 160
 ImageWaitViewer package, 280–82
 ImageWaitViewer2 package, 282
 Implicit lists, 121–23
 Information delivery technologies
 customer adoption of, 8
 innovations in, listed, 4–5
 technology diffusion curve, 8–9
 Infrared (IrDA) networks, 30, 55, 107, 268
 InputStream class, 244, 332
 InputStreamReader class, 245
 Intel, 373
 Interactive gauges, 140
 International Data Corporation (IDC), 12
 ISM band, 373
 iso-2022-jp encoding, 265

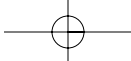


isShown() method, 113
 Item abstract class, 130–45
 Item commands, 142–44
 Item subclasses
 ChoiceGroup class, 137–39
 DateField class, 136–37
 exclusive ChoiceGroup, 137–38
 Gauge object, 139–42
 gauge states, listed, 140
 input constraints, TextField class, 133,
 134t
 interactive and noninteractive gauges, 140
 mode argument, DateField class, 136
 popup ChoiceGroup, 137–38
 StringItem and ImageItem classes,
 131–33
 text input modes, 133, 135t
 TextField class, 133–35, 136–37
 ItemCommandListener interface, 142
 ItemStateListener interface, 144

J

J2ME development tools. *See* IDEs
 (Integrated Development Environments);
 J2ME Wireless Toolkit (J2ME WTK);
 Nokia Developer's Suite (NDS) for J2ME
 J2ME Personal Profile, 37
 J2ME (Java 2 Micro Edition) technology. *See*
 also MIDP (Mobile Information Device
 Profile)
 architecture, 47–48
 CDC, 48
 CLDC, 48
 configurations, 47, 48
 KVMs, 46, 48
 optional packages, 47, 48
 profiles, 47, 48
 smart clients, developing, 15, 26
 Web Services API, 466
 J2ME Wireless Toolkit (J2ME WTK), 81–82,
 85
 J2MEUnit framework
 assert methods, 520
 TestCase class, 521–23
 TestRunner class, 525
 TestSuite, 523–24
 JAD (Java application descriptor) file, 96,
 312. *See also* MIDlet attributes

 application behavior, customizing, 484
 attributes, required versus custom, 90
 digital signatures, 90, 97
 execution triggers, registering, 73
 generating with NDS, 90–92
 MIME type association, OTA provisioning,
 98
 and OTA provisioning, 96
 property values, retrieving, 63, 90
 read-only storage, 241–42
 RMS storage required, specifying, 247
 WMA, permissions related to, 314t
 jar command, 89–90
 JAR manifest file. *See also* MIDlet attributes
 application resource files, 88
 creating, 89–90
 generating with NDS, 90–92
 MIME type association, OTA provisioning,
 98
 and OTA provisioning, 96
 property values, retrieving, 63, 90
 read-only storage, 241–42
 size limit, optimizing, 479–80
 size, matching with JAD attribute, 95
 Java AWT (Abstract Widget Toolkit), 48
 Java bootstrap classes, 87
 Java Community Process (JCP), 26, 49
 Java Debug Wire Protocol (JDWP), 511
 Java Integrated Development Environments.
 See IDEs
 Java Native Interface (JNI), 48
 Java objects serialization, 242, 244–47
 Java Portlets, 14
 Java runtime, 15, 78
 Java Specification Requests (JSRs), 49
 Java Store Procedures, 423
 Java technology
 benefits of, 45
 crossplatform, defined, 46
 garbage collectors, 45–46
 Java 2 Platform, 46
 Java Virtual Machines (JVMs), 46, 61
 main() method, 61
 open interfaces, 48–49
 WORA versus Java Everywhere, 46
 Java Virtual Machine Debugger Interface
 (JVMDI), 511
 javac utility, 87
 java.io package, 244–45
 JavaOne conferences, 46
 JavaServer Faces, 14



`javax.microedition.io` package, 261
`javax.microedition.rms` package, 247
 JBuilder Mobile Edition, 85
 JCP Executive Committees (ECs), 49
 JDBC (Java DataBase Connectivity) API, 48, 421
 JPEG images, 133
 JTWI (Java Technology for the Wireless Industry) specification, 50

K

Key codes, 149–50
 Key-event handlers, 149, 152–54
 Kilobyte Virtual Machines (KVMs), 46
 Known devices, 377. *See also* Preknown devices

L

`LayerManager`, 206, 220–21
 Layers, 200–201, 220–21
`layout` attribute, 160
 Layout management, 160–63
 Lead software, 30
 Life cycle states, 63f, 334–36
 Limited Inquiry Access Code (LIAC), 382, 392–93
`List` class, 120–25
`List` constructors, 121
`listRecordStores()` static method, 248
`listType` parameter, 121
`LocalDevice` class, 391–93
 Logic errors, 511
 Logical Link Control and Adaptation Protocol (L2CAP), 379, 381, 387

M

`main()` method, 61
 Main-window UI paradigm, 423
`Manager` class
 data stream and MIME type,
 `createPlayer()` method, 330–31
 `DataSource`, `createPlayer()` method, 332

 duration argument, 332
 media types and protocols, querying, 332
 musical notes, tone frequencies and MIDI values, 332, 333t
 playing tones, 332–33
 `SourceStream` versus `InputStream`, 332
 system properties, retrieval of device, 333, 334t
 URI locator strings, `createPlayer()` method, 329–30

manipulation argument, 187

Media capture. *See* Multimedia blog application

Media time, 337

MediaPlayer MIDlet

`Canvas`, video playback on, 354–55

 capabilities, 347

`Form`, video playback on, 352–54

 initializing players, 348

 menu, 347

 System Properties menu action, 347

 video playback, 350–52

 wav audio file playback, 348–50

 worker thread technique, 347–48

`MessageConnection` interface, 309–10

`MessagePart` class, 324–25

MIDI player MIDlet

 code, 339–42

`MIDIControl`, 346

 objectives, 338–39

`PitchControl`, 346

 player controls, 344–46

 player events, 343–44

 player initialization process, 342–43

`RateControl`, 345

`StopTimeControl`, 345

`TempoControl`, 345

`VolumeControl`, 345

MIDlet attributes

 API access permission, 103

 authentication, 100–101

 Games menu, placing MIDlet suite under, 105–06

 notification and reporting, 105, 106

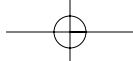
 optional informational, 100

 push-related, 100

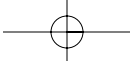
 required, 99

 size, matching JAR with JAD attribute, 95

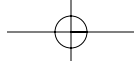
MIDlets. *See also* JAD (Java application descriptor) file; JAR manifest file; MIDlet attributes; Replaceable modules



- the AMS, 49, 61, 63–64
- API specification, 49
- compiling, 87–88
- deployment options, in the NDS, 106–07
- exiting, 69–70
- life cycle states, 63f
- MIDP Push Registry, 72–73
- MIDP UI, 64–65
- multithread programming, 73–74
- MVC pattern, 423–25
- preverification process, 88
- provisioning options, 106–07. *See also* OTA (over-the-air) provisioning
- resource files, copying, 88
- RMS storage required, specifying, 247
- sample code, minimal MIDlet, 49–50
- state changes, 61–62, 182
- states, listed, 61
- suite authentication, 100–101
- trusted versus untrusted, 90, 101–02, 312
- UI states, background and foreground, 62–63, 182
- UI thread, 74
- `MIDletStateChangeException`, 70
- MIDP (Mobile Information Device Profile). *See also* MIDlet attributes; MIDlets
 - 1.0 versus 2.0, 50
 - the AMS, 49, 61, 63–64
 - and J2ME architecture, 48
 - JBuilder Mobile Edition, 85
 - optional packages, 50, 51–52t
 - Push Registry, 72–73
 - security model, 101–02
 - UI basics, 64–65
- MIDP Generic Connection Framework specification, 72
- MIDP LCDUI. *See also* Nokia UI API
 - extensions
 - Command objects, mapping, 116–17
 - Display class, 112–13
 - Displayable class, 113–16
 - low-level API, 111–12
 - observer design pattern, 114–16
 - sample application, API demo screens, 118–20
 - screen display size, querying, 113–14
 - screen-switch, 423–24
 - title and ticker, manipulating, 114
 - MIDP LCDUI, advanced concepts
 - ARGB arrays, image conversion to/from, 165–66
 - ARGB image data format, 164
 - background state, Series 60 devices, 182
 - color schemes, specifying, 157–58
 - Command objects, mapping to soft keys, 158–60
 - commands, display order rules, 158–60
 - custom item image button example, 168–71
 - CustomItem class, 167–71
 - device controls, 156–58
 - Graphics.drawRGB() method, 166
 - image sizes, querying the best, 158
 - images, drawing parts to specified regions, 166
 - layout management, 160–63
 - layout options example, 161–63
 - NDS MIDP UI designer, 182–84
 - options menu, allocating soft keys for, 159–60
 - phone vibration, 156–57
 - pixel manipulation, 166
 - row alignment, vertical and horizontal, 160
 - row breaks, 161
 - screen backlight, flashing, 156–57
 - screen, querying physical characteristics of, 157
 - scrollable text screen example, 176–78
 - space, allocating to items and rows, 161
 - splash screens, 171–74, 192
 - text wrap, automatic, 178–81
 - transparency, 164, 165f
 - virtual Canvas, clipping and scrolling, 174–78
- MIDP LCDUI, high-level API
 - alarm alert, code, 126–27
 - Alert class, 125–28
 - Alert constructors, 125
 - alert, specifying the duration of, 126
 - AlertTypes, predefined, 125
 - Choice interface, static constants, 121
 - default command event, setting, 121–22
 - Form class, 130
 - image size, 121, 128
 - implicit lists, 121–23
 - introduced, 111
 - Item abstract class, 130–45. *See also* Item subclasses
 - Item commands, 142–44
 - item state changes, 144–45
 - List class, 120–25
 - List constructors, 121
 - listType parameter, 121



- multiple selection list, code, 124–25
- `Screen` abstract class, 120–30
- selected options, retrieval of, 122
- selection flags, setting, 122
- text box demo, code, 128–30
- text, programmatically altering, 128
- `TextBox` class, 128–30
- `TextBox` constructor, 128
- `TextBox`, retrieving information from, 128
- `TextField` class, 128
- MIDP LCDUI, low-level API, 145–46
 - anchors, listed, 147
 - animation example, 154–56, 157f
 - `Canvas` class, 145
 - clipping areas, specifying, 147
 - double-buffering, 146
 - drawing methods, `Graphics` class, 146–47
 - fonts, 148
 - game keys, 150–51
 - `Graphics` class, 146–47
 - key codes, 149–50
 - key handler example, movable text, 152–54
 - key-event handlers, 149
 - `paint()` method, 145–46, 154, 167
 - pointer events, capturing, 150, 152
 - stroke-style constants, `Graphics` class, 146
- MIDPLogger, 515–19
- MIME plugins, 13
- MIME types, MMAPI implementation, 330–31
- MMAPI (Mobile Media API). *See also*
 - `Manager` class; `Player` interface
 - architecture, 329
 - audio playback, MIDI player. *See* MIDI player MIDlet
 - `Control` interface, 337–38
 - implementation, querying, 333, 334t
 - media capture. *See* Multimedia blog application
 - media playback, audio and video. *See* `MediaPlayer` MIDlet
 - optimizing, 481–82
- MMS (Multimedia Messaging Services)
 - application interaction, 41–42
 - architecture, 41, 42f
 - asynchronous model tradeoff, 43
 - benefits and opportunities, 531
 - infrastructure, 531–33
 - message delivery sequence, 41
 - MMSC (Multimedia Messaging Service Center), 41, 531, 533, 545–46
 - mobile entertainment, 12
 - Nokia device characteristics, 536–38
 - open standard technologies, 26
 - pervasiveness of, 44
 - SMIL. *See* Synchronized Multimedia Integration Language (SMIL)
 - uses and support for, 530–31
- MMS application modes
 - originating, 535
 - originating-terminating, 535
 - peer-to-peer messaging, 534
 - terminating, 534, 535f
- Mobile advantages, 5–6
- Mobile applications
 - open standards, importance of, 25
 - social design considerations, 20
- Mobile commerce
 - browser and messaging applications, 14–15
 - consumer applications, 6
 - enterprise applications, 12–13
 - government sectors, 8
 - information delivery technologies, 4–5
 - Internet, effects of, 5
 - killer application groups, 11–13
 - sales and workplace automation, 7–8
 - strategies, 9
 - supply chain management, 6–7
 - technology diffusion curve, 8–9
 - user needs, 5
 - value chain, 10–11
- Mobile development platforms. *See also*
 - Nokia Developer Platforms
 - developer skill migration, 13–14
 - migration path issues. *See* Developer skill migration paths
 - providers, in value chain, 11
- Mobile devices
 - customization of, 5
 - limitations and solutions, 16, 17–18t
 - manufacturers, role in value chain, 11
 - market potential, 5
- Mobile enterprise applications, 12–13
- Mobile entertainment. *See also* Games
 - applications, types of, 12
 - digital content download, 6
 - digital content downloads, 12
- Mobile Media API Technical Note, Forum
 - Nokia, 331
- Mobile networks. *See* Wireless networking technologies
- Mobile value chain, 10–11
- Mobile workers

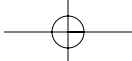


enterprise tasks of, 13
 number of, in U.S. and Europe, 12
 Model-View-Controller (MVC) pattern,
 423–25. *See also* Trivia game example
 Moore's law, 8, 25
 Motorola, 53
 move() method, 200–201
 MP3 players, 6
 Multimedia blog application
 audio capture, 361–63
 blog entries, submitting, 363–64
 blog servlet, 364–68
 camera viewfinder, Series 60 emulator, 357,
 359f
 image capture, 357–61
 process overview, 355–56
 MultipartMessage class, 324–25
 Multiplayer games, 12
 Multiple selection lists, 124–25
 Multithread programming, 73–74

N

NetBeans open source project, 85
 Network traffic monitor, 277–78
 newMessage factory method, 310
 Nokia
 Bluetooth SIG (Special Interest Group), 373
 mobile sales solutions, 7
 sniff interval, 377
 Nokia Connectivity Framework, 315
 Nokia Developer Platforms. *See also* Series
 40 Developer Platform; Series 60
 Developer Platform
 application certification, 56
 architecture, 27–28
 develop-and-optimize approach, 27–28. *See
 also* Develop-and-optimize process
 developer skill migration, 14
 devices, number sold in 2004, 24, 55–56
 four technology pillars, 24
 Java developer's perspective, 27f
 as market leader, 55–56
 marketing assistance, 56
 open standard technologies, listed, 26
 pre- and post-2004 released devices, 28
 Series 80, 37
 Series 90, 38, 53–54
 support and resources, 56

thin-client application paradigm, 44
 Nokia Developer's Suite (NDS) for J2ME
 deployment options, MIDlet suites, 106–07
 digital signatures, 100–101, 103, 104f
 downloading and installing, 78
 emulators, testing and running with, 92
 IDE integrated edition, 78
 JAR manifest and JAD files, generating,
 90–92
 JDK software requirement, 78
 major features, 79–80
 OTA process, testing, 97
 SDKs, pre-installed, 80. *See also* SDKs
 standalone application, 78
 viewport, 184, 185f
 visual UI designer, 182–84
 Windows and Linux editions, 78
 Nokia Developer's Suite (NDS) for MMS,
 542–45
 Nokia devices. *See also* Series 40 devices;
 Series 60 devices
 6800, 13
 7700 multimedia phone, 12
 advanced controls, 156–58
 Communicators, 13, 37, 54
 computing power of, 8
 JPEG image support, 133
 MMS, characteristics for, 536–38
 N-Gage game deck, 12, 15, 35
 PNG image support, 133
 Series 80, 15
 Series 90, 15
 system properties, retrieval of, 333, 334t
 Nokia Mobile Server Services Library
 connection configuration, Nokia API, 547–48
 connection configuration, SAMS API, 548
 instantiating the driver, Nokia API, 546–47
 instantiating the driver, SAMS API, 547
 MMSC, direct access via Java API calls,
 545–46
 receiving messages, Nokia API, 555
 receiving messages, SAMS API, 556–57
 simple messaging, Nokia API, 549–50
 SMIL message, Nokia API, 550–51
 SMIL message, SAMS API, 551–53
 SMS message, SAMS API, 553–54
 Nokia Networks, 13
 Nokia Tradepoint program, 56
 Nokia UI API extensions
 DeviceControl utility class, 186
 DirectGraphics class, 186–87



DirectUtils utility class, 186
 drawPixels() method, 187
 drawPolygon() and fillPolygon()
 methods, 187
 FullCanvas class, 184–86
 getPixels() method, 187
 manipulation argument, 187
 Noninteractive gauge, 140, 279–80
 Normal mode, 377
 Notification and reporting MIDlet attributes,
 105, 106
 notifyDestroyed() method, 62, 64, 70
 notifyPaused() method, 62, 63

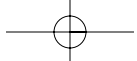
O

Object Exchange Protocol (OBEX), 380, 381
 Object management. *See also* Trivia game
 example
 back screen stack, 447
 factory methods, 439
 object pools, 443–44
 Singletons, 439
 static class, 434–38
 Object Push Profile (OPP), 381
 Occasionally connected application
 paradigm, 417
 OMA Client Provisioning, 26
 Open Mobile Alliance (OMA) standard, 26
 openDataInputStream() method, 264, 266,
 268
 openDataOutputStream() method, 266,
 268
 openRecordStore() method, 248, 249
 Optimizelt, 85
 Optional informational MIDlet attributes, 100
 Options menu, allocating soft keys, 159–60
 Originating mode, MMS, 535
 Originating-terminating mode, MMS, 535
 OTA (over-the-air) provisioning. *See also*
 MIDlet attributes
 installation events, notification and
 reporting of, 103, 105
 MIME type association, JAD/JAR files, 98
 NDS, testing process using, 97
 process steps, 96–97
 server setup, 98
 Otis, 7

OutputStream class, 244
 OutputStreamReader class, 245

P

Packet-switch networks (2.5G networks), 16
 paint() method, 145–46, 154, 167, 176
 Paired devices, 377
 Panasonic, 53
 Park mode, 377
 Path directory separators, 94
 pauseApp() method, 62, 63
 Paused state, for MIDlets, 61
 PBP (Personal Basis Profile), 48
 Peer-to-peer messaging mode, MMS, 534
 Persistent storage, 241–42
 Photo Viewer example
 automated build with Ant, 93–96
 automatic slide show, running, 73–77
 code, 66–68, 70–72, 76–77
 compiling, 87–88
 data serialization, 246–47
 End key versus Exit command, 69–70
 exiting, 69–70
 file structure and subdirectories, 86–87
 the JAD file, 90–92
 JAR manifest and JAD files, generating,
 90–92
 MIDlet constructor, 69
 multithreading, 73–74
 packaging, 89–90
 preverifying, 88
 primary functionality, 65
 the process, 65, 66f, 69
 project content, prior to building, 87
 record store, storing ImageAttribute data
 in, 250–51
 resource files, copying, 88
 search and sort, 255–56
 test and run, 92
 Timer class, 75–76
 TimerTask class, 75
 TimerTask objects, scheduling, 75–76
 Photo Viewer example, networked
 animated transit screen, 284–87
 application screen, pictured, 277f
 FetchWorker, 273–77, 292–94
 HTTP cookies, 288–92

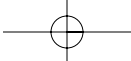


HTTP header handlers, 296–99
 HTTP protocol, use of, 269
 noninteractive gauge, 279–80
 objectives, 269
 PhotoServlet, 270–72, 288–89
 PhotoViewer, 272–73
 servlet/MIDlet communication, 269
 session based photo viewer, pictured, 295f
 still-image transit screen, 280–82
 still-image transit screen, improved, 282–84
 tight coupling, 269–70
 Piconets. *See* Bluetooth technology, piconets
 Pixel manipulation, 166
 Pixel refresh rate, 223
 Pixel-level collision, 203–04
 platformRequest() method, 64
 Player interface
 event strings, 336
 implicit state changes, 336
 life cycle states, 334–36
 loops, setting number of, 337
 media files, querying the status of, 336–37
 media time, 337
 PlayerListener interface, 336
 resource consumption, 334
 synchronizing players, 337
 Player.close() method, 335
 Player.deallocate() method, 335
 PlayerListener interface, 336
 Player.prefetch() method, 335
 Player.realize() method, 335
 Player.start() method, 335
 Player.stop() method, 335
 Pleumann, Jörg, 490
 PNG images, 133, 235
 Pointer events, capturing, 150, 152
 Postprocessing, 499–500
 PP (Personal Profile), 48
 Preknown devices, 377, 394
 Preprocessing and postprocessing, 488–89.
 See also Antenna
 Preprocessor directives, 492, 493t
 Preverification process, 88
 processAlpha argument, 165
 Profiles, 47, 48, 373, 381
 Psion, 53
 Public keys, 301–02
 Push Access Protocol (PAP), 572
 Push Proxy Gateway (PPG), 571–72

PushRegistry class, 72–73
 Push-related MIDlet attribute, 100

R

Random Access Memory (RAM), 241
 Rapid Application Development (RAD) tools, 15
 Read-only storage, JAD and JAR manifest files, 241–42
 readXXX() method, 245
 RecordComparator, 254–55
 RecordEnumeration, 252–53
 RecordFilter, 253–54
 RecordListener interface, 251–52
 RecordStore. *See also* Photo Viewer
 example
 browsing, using RecordEnumeration, 252–53
 closing, 248, 249
 creating, 247–48
 debugging output, redirecting with MIDPLogger, 516
 filtering, using RecordFilter, 253–54
 listeners, 251–52
 listing stores, 248
 record IDs, 249–50
 records, manipulating, 249
 shared, creating and accessing, 248
 size limitations, 247
 size, version, and last modified date, querying, 249
 sorting, using RecordComparator, 254–55
 Rectangle collision, 203–04
 Reference pixels, 209
 Refresh rate, 223
 registerAlarm() method, 72
 registerConnection() method, 72–73
 Remote storage, 242
 RemoteDevice class, 394
 repaint() method, 145, 154, 167
 Replaceable modules
 the build workflow, 487f
 class, replacing part of, 487–88
 customized JAD files, 484
 purpose for using, 483–84
 resource file modules, 484–85
 source code modules, 485–87



Required MIDlet attributes, 99
`resumeRequest()` method, 62, 63
 RFCOMM protocol, 380, 381
 Ringtones, 12
 RMI (Remote Method Invocation) Optional Package, 48
 RMS (Record Management System), 242, 247, 480. *See also* `RecordStore`
 Row alignment, vertical and horizontal, 160
 RPC protocol, 456, 460
`run()` method, 75
 Runtime errors, 511

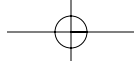
S

`safeShutdown()` method, 70
 Samsung, 53
 Sanyo, 53
 SAP, 7
`scanlen` argument, 165
 Scatternets, 375–76
`schedule()` method, 76
`scheduleAtFixedRate()` method, 76
`scheduledExecutionTime()` method, 75
 Screen abstract class, 120–30
 Screen-switch navigation model, 423–24
 Screen-switch paradigm, 423
 Scrolling, 174–78
 SDKs
 concept SDKs, 81
 installing, 80–81
 internal file structure, 82–83
 JBuilder Mobile Edition, 85
 network traffic monitor, 277–78

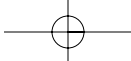
```
verify
```

 program, 88
 SecureConnection interface, 267
 SecurityInfo interface, 302
 SELECT_COMMAND, 121
 Sendo, 53
 September 11, 2001, 8
 Serial port connectivity, 268
 Serial Port Profile (SPP), 381
 Serialization, Java objects, 242, 244–47
 Series 40 Developer Platform
 devices covered, 28
 enabler technologies and APIs supported, 30
 font sizes, standard Latin, 148

J2ME technology, use of, 15, 30
 lead software, installing, 30
 native client applications, 29–30
 network traffic monitor, 277–78
 Nokia Connectivity Framework, 315
 software stack, 29–30
 target market, 28
 Series 40 devices. *See also* MIDI player
 MIDlet
 alert screen, 127f
 Bluetooth stack, activating, 385f
 browser, launching the, 64
 connectivity protocol support, 30
 exclusive `ChoiceGroup`, 138f
 extensions, 30
 gauges, 141f
 implicit list, 122f
 individual elements, interaction with, 32, 34f
 Item command for the `StringItem`, 143f
 LCD display, 30
 menu hierarchy, 32, 33f
 MIME types supported, 331
 multiple selection list, 124f
 Options soft key, use of, 33f
 pictured, 31f
 popup `ChoiceGroup`, 138f
 RMS storage, 247
 scrollable text screen, 177f
 StringItem and ImageItem, 132f
 text wrap, automatic, 181f
 TextField and DateField, 135f
 title and ticker, 114f
 user interface, 32–34
 Series 60 Developer Platform
 versus Series 60 Platform, 35
 software stack, 35
 Symbian OS C++ technology, use of, 15, 35, 53
 Series 60 devices. *See also* MediaPlayer
 MIDlet; Multimedia blog application
 alert screen, 127f
 background MIDlet, 182
 browser, launching the, 64
 data storage, 36
 exclusive `ChoiceGroup`, 138f
 gauges, 141f
 GIF image support, 133
 implicit list, 122f
 keypad-based, Symbian, 53



- multiple selection list, 124f
- numbers sold in 2004, 34
- pictured, 36
- popup *ChoiceGroup*, 138f
- RMS storage, 247
- scrollable text screen, 177f
- StringItem* and *ImageItem*, 132f
- text wrap, automatic, 181f
- TextField* and *DateField*, 135f
- title and ticker, 114f
- user interface, 35–36
- Series 80 Developer Platform, 37
- Series 90 Developer Platform, 38, 53–54
- Server API for Mobile Services (SAMS), 546
- ServerSocketConnection* interface, 267
- Service Discovery Database (SDDB), 395–97
- Service Discovery Protocol (SDP), 379, 381
- Service Indication, 572
- Service Loading, 572–73
- Service patterns, 397
- Service-Oriented Architecture (SOA), 307
- ServiceRecord* class, 396
- serviceRepaints()* method, 145
- Session key, 302
- SessionViewer* package, 287
- setAnimatedTile()* method, 217, 220
- setCell()* method, 215, 216
- set-cookie header, 288, 289
- setCurrent()* method, 64, 113
- setFrameSequence()* method, 208
- setFullScreenMode()* (Boolean) method, 197
- setInitialInputMode()* method, 133
- setPosition()* method, 200
- setSelectCommand()* method, 121
- setSocketOption()* method, 266
- setStaticTileSet()* method, 215
- setTimeout()* method, 125–26
- setTransform()* method, 209, 210
- Short range radio networks, 16, 18
- showImage()* method, 69
- Siemens, 53
- Singleton objects, 46, 112, 391
- Singleton pattern, 439
- Slots, 373
- Smart linking, 499–500
- SMS (Short Message Service), 13, 308, 309, 315–17
- Sniff mode, 377
- SocketConnection* interface, 266
- Soft keys, mapping, 158–60
- Sony Ericsson P800, 53–54
- SourceStream* class, 332
- Splash screens, 171–74, 191, 192
- SplashScreenThread* class, 171–72, 174
- Sprites
 - animation, 207–09
 - collision detection, 203–04
 - constructors, 202
 - frame sequencing, 208
 - frames, numbering and storage, 202, 203f, 207–09
 - images, loading, 212
 - nonanimated versus animated, 202
 - reference pixels, 209
 - sequence length versus frame count, 208
 - transforms, 209–10, 211t
- stackmap* attribute, 88
- startApp()* method, 62, 66, 69, 171–72
- StarTeam, 85
- Static factory methods, 261, 439
- Still-image transit screen, 280–84
- StringItem* class, 131–33
- Stroke-style constants, *Graphics* class, 146
- Sun Java Studio, 85
- Sun Microsystems, 46, 56, 292
- Supply chain management, 6–7
- Symbian OS C++ technology
 - API, 53
 - application utility layer, 55
 - architecture, 54–55
 - base system APIs, 55
 - communications, 55
 - evolution of, 53
 - GUI framework and services, 55
 - keyboard-based, 54
 - keypad-based, 53
 - low-level device features, accessing, 15
 - N-Gage game deck games, 35
 - as open standard, 26
 - OS, described, 53
 - partners and licensees, 53
 - pen-based, 53–54
 - product lines, 53–54
- Synchronization pattern, 460–61
- Synchronized Multimedia Integration Language (SMIL)
 - capabilities and functions of, 43, 538, 539f
 - document structure, 539
 - layout management, 540–42



- media elements, 540, 541t
- music slide show example, 542
- playback control elements, 539–40
- transition effects, 542
- SyncML, 14
- `System.gc()`, 46, 223, 241
- `System.out.println()` method, 513–14

T

- Technology diffusion curve, 8–9
- Telephony Control Protocol Specification (TCS binary), 380
- Terminating mode, MMS, 534, 535f
- `TestCase` class, 521–23
- `TestRunner` class, 525
- `TestSuite`, 523–24
- Text wrap, automatic, 178–81
- `TextBox` class, 128–30
- `TextField` class, 128, 133–35, 136–37
- `TextMessage` interface, 308–09
- Thawte, 101, 302
- Thin-client application paradigm, 44, 560–61
- Third-Generation Partnership Project (3GPP), 26
- Thread, defined, 74
- `TiledLayer`
 - animation, slowing, 220
 - background images, creating, 213, 215f
 - cell animation, 216–17
 - constructor, 214–15
 - source image numbering, 214
 - tile number versus index number, 214, 216
 - tile placement in cells, 215
- `tileWidth` and `tileHeight` parameters, 214
- Time-Division Duplex (TDD), 373
- `Timer` class, 75–76
- `TimerTask` class, 75
- `TimerTask` objects, 75–76
- TogetherSoft, 85
- Toshiba, 373
- `transform` argument, 166
- Transit screens
 - animated, 284–87
 - need for, 278–79
 - noninteractive gauge, 279–80
 - still-image, 280–84
- Transparency, 164, 165f, 235

- Trivia game example
 - back screen stack, implementing, 447–49
 - backend database setup, 420–23
 - chat messaging, 419, 420f
 - client/backend synchronization, history menu, 417–19
 - the controller, 428–29
 - database structure, 422
 - directive usage, Antenna, 494–97
 - implementation, selecting at runtime, 443
 - menu options, 415, 416f
 - the model, 429–34
 - multimedia content, storage of, 422
 - multiple implementations, selecting from, 442–43
 - MVC pattern implementation, 425–34
 - object pools, 444–46
 - RPC transaction, client side, 456–58
 - RPC transaction, server side, 458–60
 - search sessions, Google, 419
 - Singleton pattern, 439–42
 - stateful static class, 435–36
 - static class, requiring initialization, 436–39
 - synchronization, client side, 461–64
 - synchronization, server side, 464–66
 - trivia questions, types of, 415–17, 418f
 - the view screens, 425–28
 - Web Services gateway, client side, 467–68
 - Web Services gateway, server side, 468–73
 - worker thread implementation, 454–56

U

- `UDPDatagramConnection` interface, 268–69
- UI testing guidelines, 526–27
- UIQ devices, 53–54
- Understanding Web Services: XML, WSDL, SOAP, and UDDI (Newcomer), 466
- Unit testing, 519–20. *See also* J2MEUnit framework
- Universal Mobile Telecommunications system (UMTS) networks, 16, 30
- UPS package tracking, 7
- URI locator strings, 329–30
- URL schemes, 261–62, 309–10
- USB networks, 30, 55
- UTF-8 encoding, 245, 265
- UUIDs (universally unique identifiers), 387–88, 388–89t

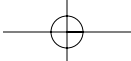
V

Value chain, mobile, 10–11
 Verisign, 101, 302
 Verizon, 12
 Viewport, 184, 185f
 Vodafone, 12
 VPN (virtual private network), 13

W

WaitScreen class, 453–54
 Wallet application, 574–75
 WAP browsers/gateways, 572–73
 <card> tag, 40, 41f
 application model, 39
 CSS (cascading style sheets) support, 41
 dual mode, 41
 network architecture, 39–40, 561
 and OTA provisioning, 98
 pervasiveness of, 44
 protocol stack, 562
 Push Access Protocol (PAP), 572
 Push Proxy Gateway (PPG), 571–72
 Service Indication, 572
 Service Loading, 572–73
 wireless profiled TCP/IP stack, 563–64
 Wireless Session Protocol (WSP), 563
 Wireless Telephony Applications Interface (WTAI), 573–74
 Wireless Transport Layer Security (WTLS) protocol, 563
 WML, 26, 40, 562, 565–67
 WMLScript, 567
 XHTML MP, 40–41
 Web Services gateway, 466, 467f
 WiFi (wireless fidelity) networks, 16
 WiFi (wireless fidelity) providers, 10
 Wireless Application Environment (WAE), 562
 Wireless carriers, 10, 12
 Wireless Markup Language (WML), 26, 40, 562, 565–67
 Wireless Messaging API (WMA). *See also* Chat example application
 access permissions, 312, 314t
 concatenation, SMS messages in GSM networks, 309

 connection interfaces, 310, 311f
 enterprise messaging middleware, 307
 versus HTTP, 306–07
 MessageConnection interface, 309–10
 methods, MessageConnection interface, 310
 MIDlets, trusted versus untrusted, 312
 MMS capabilities, 323
 mobile messaging, advantages, 307
 MultipartMessage and MessagePart classes, 324–25
 multiple recipients, 311
 new features, version 2.0, 323–25
 push registry, message listener in, 313–14
 receiving messages, 312
 restricted ports, 313
 security, 312–13
 sending messages, 310–11
 server mode connection, 311
 SMS emulation, 315–17
 SMS, uses with smart clients, 308
 TextMessage and BinaryMessage interfaces, 308–09
 URL schemes, 309–10, 323–24
 versions, 308
 Wireless networking technologies
 limitations and solutions, 18–19
 listed, 16, 18
 Wireless profiled TCP/IP stack, 563–64
 Wireless Session Protocol (WSP), 563
 Wireless Telephony Applications Interface (WTAI), 573–74
 Wireless Transport Layer Security (WTLS) protocol, 563
 WMLScript, 567
 WORA (Write Once, Run Anywhere), 46
 Worker threads. *See also* Generic visual thread framework; Trivia game example
 main application, hanging, 267
 MediaPlayer MIDlet, 347–48
 Photo Viewer example, 273–77
 preemptive starting of, 279
 and transit screen display, 282–83
 WorkerRunnable interface, 452
 WorkerThread class, 452
 World Wide Web Consortium (W3C), 26, 40, 43, 264
 writeXXX() method, 245
 wtkobfuscate task, 500–501
 wtkpreprocess task, 492, 497–99
 wtksmartlink task, 499–500



594

Index

X

Xerox, 7

XHTML Mobile Profile (MP), 26, 40–41,
565–67

XML Web Services, 13

XML/SOAP API, 26, 466

Z

Z-order, 220

