

# Index

## A

- abld.bat, 6–7, 12, 31, 47–48
  - creating, 35
- Abstract base classes, 80
- Active Objects, 129–131
  - checklist, 131–132
  - common pitfalls, 142–143
  - defined, 129, 705
  - implementing, 131–132
  - practical example, 132–142
    - canceling an outstanding request, 138–139
    - construction, 133–134
    - destructor, 139
    - RunError(), handling errors in, 137–138
    - RunL(), 135–137
    - starting the Active Object, 134–135
    - starting the Active Scheduler, 139–142
  - and responsiveness, 220
- Active Scheduler, 128–129
  - defined, 705
  - instantiation, 139–140
  - lifecycle of, 140–141
  - starting, 139–142
- Add Field, 304, 306
- Advanced application deployment and build
  - guide, 27, 54, 67–72
  - ARM targets, building, 70–72
  - device identification:
    - during execution, 70
    - at install time, 69
  - device identification UIDs, 68–69
  - machine UIDs, 69, 713
  - platform UIDs, 67–68
  - resource file versions and compression, 70
- Advanced communication technologies, 489–551
  - HTTP, 490, 491–501
  - messaging, 490, 508–543
  - telephony, 490, 543–551
  - WAP (Wireless Application Protocol), 490, 502–508
- Agenda Server, 639–640
- Agent APIs, 618
- AIF**, 33
- AIF Builder, 14, 705
- .aif files, 43–46
  - aifbuilder tool, 43, 705
  - aiftool utility, 43
  - captions, 45–46
  - file localization, 44
  - icons, 44
  - MIME support, 44–45
- aif folder, 38
- aifbuilder tool, 705
- AKN\_LAF\_COLOR(), 565
  - color palette values for macro, 566
- AknsDrawUtils, 239
- AknsUtils, 239
- AlarmOrganiser example application, 619
- Alloc(), 100
- AllocL(), 95, 100
- AllocLC(), 100
- AllocReadResourceL(), 568
- Animation, 554–555, 582–615
  - architecture, 582–585
  - client-side approach to, 586–588
  - decoding, 595–597
  - Series 60 1.x, 595–596

- Animation, *continued*
  - Series 60 2.x, 596–597
  - direct screen access (DSA), 588–594
    - architecture, 589–590
    - implementation, 593–594
    - key classes for, 590–593
  - double buffering, 585–586
  - encoding, 597–601
    - Series 60 1.x, 598–599
    - Series 60 2.x, 599–601
  - image conversion, 594–601
  - image manipulation, 594–604
  - image rotation, 601–602
    - Series 60 1.x, 601
    - Series 60 2.x, 601–602
  - image scaling, 602–604
    - Series 60 1.x, 602
    - Series 60 2.x, 602–603
  - offscreen bitmaps, 585–586
  - resource components, 583
- Animation** example application, 555, 585
- AnsPhone** example application, 54, 491, 544, 546, 550, 555, 604–606
- .app filename extension, 36
- AppArc** (application architecture), 44, 176–177
- Append(), 101
- Application architecture, 174, 178–201
  - application initialization, 179–182
  - appropriate architecture, choosing, 197–200
  - AppUi methods, 182–183
  - Avkon view-switching architecture, 191–197
    - when to use, 197–198
  - core application classes, 178–179
  - designing an application UI, 183–184
    - view terminology, 183–184
  - dialog-based architecture, 188–191
    - when to use, 199–200
  - file handling, 200–201
  - traditional Symbian OS control-based architecture, 184–188
    - when to use, 198
- Application deployment, 26
- Application design, 173–221
  - application architecture, 174, 178–201
  - application framework, 174, 175–177
  - ECom, 174, 206–213
  - good application behavior, 174
  - internationalization, 174
  - splitting the application UI and the engine, 174, 201–206
- Application engines, 618, 625–649
  - calendar engine access, 639–646
    - adding, editing and deleting entries, 643–644
    - agenda entries, reading, 641–642
    - agenda models/concepts, 639
  - Agenda Server, 639–640
  - anniversaries, 639
  - appointments, 639
  - calendar alarms, 645–646
  - entry model, 639
  - events, 639
  - indexed model, 639
  - instance model, 639
  - notification of changes and progress monitoring, 640–641
  - searching instances and filtering, 643
  - to-dos, 639
- Camera APIs, 627–632
  - Camera Server, 628–629
  - changes in Series 60 2.x, 630–632
  - settings, 630
  - taking a picture, 629
  - using with an emulator, 628
- compact business cards/vCards, 638
- log engine, 625–627
  - log entries, reading, 625–627
- phonebook engine, 632–638
  - creating and accessing phonebook items, 632–635
  - phonebook searching, 636–637
  - receiving notification, 635–636
  - speed dialing and common dialogs, 637–638
- photo album engine, 646–649
  - photo album folders, accessing, 646–648
  - photo album UI components, 648–649
- Application Framework, 555, 557, 705
- Application Launcher, 33, 40–41, 44–45, 178, 705
- Application panics, in Series 60 emulator, 67
- Application programming interface (API), 5–6, 76, 78, 97, 705
- Application UI components, 223–287
  - ContextMenu** application, 225
  - controls, 224, 225–237
  - event handling, 224, 239–244
  - forms, 290, 302–310
  - menus, 224, 254–264
  - NavigationPane** application, 225
  - panes, 224, 264–286
  - resource files, 224, 245–254
  - SimpleMenu** application, 224
  - skins, 224, 237–239
  - TitlePane** application, 225
- Application-initiated redraws, 243
- AppUi(), 568
- AppUiFactory(), iEikonEnv, 263
- Arcs, 576
- ARM processor, 20, 706
- ARM targets:
  - building, 70–72
  - building and freezing DLLs, 71–72
  - function exports, 70–71
  - writable static data in DLLs, 71
- ARM4, 6, 20, 706
- ARM1, 6, 20, 706
- Ascent (metric), 571

ASCII (American Standard Code for Information Interchange), 101, 104, 150, 706

Assertions, 661–662  
side effects of, 662

Associated granularity, `CArray` types, 117

Asynchronous function handling, Series 60 1.x vs. Series 60 2.x, 603

Asynchronous services, 77  
Active Objects, 129–131  
Active Scheduler, 128–129

Audio, 555, 604–614  
audio data, 610–612  
audio device modes, 607–608  
device modes, 607–608  
.midi format, 604  
recording, 605–609  
Series 60 audio libraries, 604  
streaming, 612–614  
tones, 609–610  
.wav format, 604

**Audio** example application, 604, 611

**AudioPlayer** example application, 555

Authorization, 497  
`autoexp.dat`, 63

Automatic variables, 33

Avkon, 45, 706  
*Avkon Skins User's Guide*, 239  
Avkon view-switching architecture, 191–197  
Avkon View-Switching Architecture, 619  
when to use, 197–198  
`AVKON_NOTE` control, 314–315  
`avkon.hrh`, 41

## B

Back key, 304

Baseline (metric), 571

Basic drawing, 554, 559–569  
color and display modes, 565–567  
graphics devices and graphics contexts, 562–565  
pens and brushes, 567  
screen coordinates and geometry, 562

Basic variable types, 76, 82–83

**BasicDrawing** example application, 555, 560, 563, 565, 567

BC, *See* Binary compatibility (BC)

Bearer, 465, 487, 502–504, 507, 706

Binary compatibility (BC), 71–72, 706

Binary switch setting item reference, 390

`BitBltMasked()`, 580–581

BITGDI library, 559

**Bitmap** example application, 555, 580

`BITMAP...END` command, 578

Bitmaps, 554, 578–581  
functions, 581  
generating for application use, 578–580  
loading/drawing, 580  
masking, 580–581  
offscreen, 585–586  
START BITMAP commands, 579

Black-box testing, 667, 706

`bld.inf`, 6–8, 13, 30

`bldmake` command, 30

Blit, 586, 706

Bluetooth (BT), 22, 50, 430–431, 434, 438, 446, 471, 706  
baseband, 471  
Bluetooth client, 486  
Bluetooth Security Manager, 477–479  
Bluetooth Server, 485–486  
**BluetoothCHAT** application, 473–486  
device and service discovery, 479  
DUN (Dial-Up Networking) Profile, 472  
Fax Profile, 472  
File Transfer Profile, 472  
Hands Free Profile, 472  
Headset Profile, 472  
Link Manager Protocol (LMP), 471  
Logical Link Control and Adaptation Protocol (L2CAP), 471  
Object Push Profile, 472  
physical layer, 471  
profiles, 471, 472  
protocols, 471–472  
Radio Frequency Communications (RFCOMM), 472  
`RHostResolver`, 479  
`RNotifier`, device discovery using, 479–480  
security, 477–479  
Serial Port Profile, 472  
Service Advertisement, 472–477  
service discovery, 481–484  
Service Discovery Database (SDD), 472–473, 474  
closing, 477  
Service Discovery Protocol (SDP), 472  
socket communication, 484–486  
stack, 471  
**BluetoothCHAT** application, 431, 473–486  
Service Advertisement, 474–477  
Service Discovery Database (SDD), closing, 477  
service record attributes, creating, 475–476  
service records, removing, 476–477

`bmconv`, 58–60, 472, 578, 706

BMP (bitmap), 559, 579, 706

Body part, resource files, 247, 249–253

Boolean, 82, 83, 507, 706

Borland C++ Builder 6 Mobile Edition, 6, 671

Borland C++ BuilderX Mobile, 6, 8, 671

Borland C++ IDE, 9

Boundary-value analysis, 667

`BUF`, 254  
`BUF<n>`, 254  
`BUFS`, 254  
`BullseyeCoverage`, 672  
`BYTE`, 254

## C

C++ namespaces, 81

CA (certificate authority), 706

CActiveScheduler, 87

CActiveScheduler::Start() function, 140–142

CAknAppUi, 258

CAknConfirmationNote, 311

CAknDialog, 291

CAknErrorNote, 311

CAknGlobalNote, 312

CAknInformationNote, 311

CAknNoteDialog, 312

CAknProgressDialog, 312

CAknView, 258

MenuBar(), 263

CAknWaitDialog, 311, 317

CAknWaitNoteWrapper, 311

CAknWarningNote, 311

Calendar engine access, 639–646

adding, editing and deleting entries, 643–644

agenda entries, reading, 641–642

agenda models/concepts, 639

Agenda Server, 639–640

anniversaries, 639

appointments, 639

calendar alarms, 645–646

entry model, 639

events, 639

indexed model, 639

instance model, 639

notification of changes and progress monitoring, 640–641

searching instances and filtering, 643

to-dos, 639

Calendar view switching, 621–622

**CallSummary** example application, 619

Camera APIs, 627–632

Camera Server, 628–629

changes in Series 60 2.x, 630–632

settings, 630

taking a picture, 629

using with an emulator, 628

Camera view switching, 622

Cancel(), 129–130, 142

CancelNoteL(), 325

**CArray** types, 117–126

associated granularity, 117

basic APIs, 118–121

descriptor arrays, 125–126

finding, 124–125

limitations, 126

sorting, 121–124

CBitmapBitBltMaskContainer::Draw(), 580–581

CBitmapDevice, 563

CBufStore, 158

**C-classes**, 79

CCoeControl, 567–569

CCoeEnv, 573

CCsvFileLoader() function, CCsvFileLoader, 149

CCustomCtrlDlgCustomControl, 300–302

CDirectFileStore, 158–160

CDirectScreenAccess, 589, 591–592

CEikonEnv, 573

CElementEngine instance, 93–94, 140

CElementList, 92

CElementsEngine, 80–81

CEmbeddedStore, 158

**Certificate Generator**, 50–52, 706

CFbsBitGc, 563

CFbsBitmap, 578

CFbsBitmapDevice, 563

CFbsDevice, 563

CFileStore, 158–159

CFont, 571

CGraphicsDevice, 563

.chm files, 66

Cipher suites, 453–454, 706

default order of, 455

strong cryptography, 454

weak cryptography, 454

Class variants, 662–663

Cleanup stack, 84, 89–91, 707

advanced use of, 96–97

non-CBase-derived heap-allocated objects, 96

**R-classes**, 96–97

CleanupClosePushL(), 96–97, 145

CleanupDeletePushL(), 96

CleanupReleasePushL(), 96–97

CleanupStack::PopAndDestroy(), 145

CleanupStack::PushL(), 87

Client, 79, 165, 166, 707

Client MTM API, 512–526

capabilities, setting, 527–528

connecting a session to the messaging server, 514–516

constructing, 516–517

CSendAppUi class, 531–539

CreateGeneralMessageL(), 537–538

creating a message, 535–536

dynamic menus, 533–534

getting started, 532–533

HandleSendAppUiCommandL(), 538–539

handling commands, 534–535

**MMS MTM:**

adding attachments, 524–526

creating the message, 523–524

sending an entry, 526

using, 523–526

validating an entry, 526

**MTM**, choosing, 528–530

**Send-As API**, 526–530

Send-As object, creating, 527

**SMS MTM**, using, 517–523

**UIDs** (unique identifiers), 516

- using, 514–526
- Client pull, 503
- ClientAnimation** example application, 555, 586–587
- Client/server architecture, 77, 165–170
  - server sessions, 165–166
    - and Inter-Process Communication (IPC), 166–168
    - server review, 169
    - subsessions, 169–170
- Client-side handle, 79
- clindex, 63
- Clipping, 568
- Clipping region, 707
- Close(), 144–145, 148
- CMSvEntry, 511
- CMSvEntrySelection, 512
- CMSvOperation, 512
- CMSvSession, 511
- CMSvStore, 511
- Codec (coder/decoder), 504, 596, 598, 604, 606, 707
- Coding standards, 81, 659–661, 670–671, 683, 696, 707
- Collection classes, 76, 109–126
  - CArray** types, 117–126
    - associated granularity, 117
    - basic APIs, 118–121
    - descriptor arrays, 125–126
    - finding, 124–125
    - limitations, 126
    - sorting, 121–124
  - RArray** and **RPointerArray** types, 110–117
    - basic APIs, 110–113
    - finding, 115–117
    - sorting, 113–115
- Color and display modes, 565–567
- COM, 66, 472, 693, 707
- Command line building, 9
- Command-line tools, IDE vs., 8–9
- Comma-separated value (CSV), 77
- CommDB, 66, 458–459, 707
- Comms module (CSY), 432, 707
- Comms Server (C32), 431
  - connecting to, 434
  - starting, 434
- Communications fundamentals, 427–487
  - Bluetooth (BT), 431
  - infrared, 431
  - serial communication, 430, 431–438
  - sockets, 430, 438–457
  - TCP/IP (Transmission Control Protocol/Internet Protocol), 430, 457–465
- Compact business cards, 638, 707
- Compact business cards/vCards, 638
- Competence centers—Series 60, 6, 707
- ComponentControl(), 243
- Compound controls, 225–228
  - creating, 232–236
- Compulsory skin-providing controls, 238
- CONE (Control Environment), 144, 176, 707
- Confirmation note, 311
- Confirmation queries, 326, 327
- ConfirmationNote** application, 313–315
- Connect(), 144–145
- Connected WSP session, 504–505
- Connection, 497
- Connectionless WSP, 505–506
- Console applications, 27, 672, 707
- ConstructL(), 94–95
- Container controls, 184, 707
- Content-Length**, 497
- Context pane, 265, 272–274
  - basics, 273–274
  - defined, 272–274
  - displaying an image in, using resources, 274
- ContextMenu** example application, 225, 262–264
- Context-sensitive menus, 262–264, 707
  - defining using resources, 262
  - displaying, 262–264
- Control Environment (CONE), 144, 176, 707
  - functions, 568
- Control stack, 241
- Controls, 224, 225–237, 707
  - compound, 225–228
    - creating, 232–236
    - container, 184, 707
    - custom, in dialogs, 300–302
    - establishing relationships between, 236–237
    - simple, 225–228
      - creating, 229–232
      - using menus in, 258
    - window ownership, 228–229
    - and windows, 225
- Controls** example application, 224, 229, 233, 248
- Cooperative multitasking, 127
- Copy(), 101
- CountComponentControls(), 243
- CPermanentFileStore, 158–159
- CPersistentStore, 158–159
- cpp, 9
- CPrinterDevice, 563
- CreateGraphicsContext() function, 563
- CreateSession(), 166
- CreateWindowL(), 567
- Crop, 349, 707
- CSecureStore, 157, 158
- CSendAppUi** API, 513–514
- CSendAppUi** class, 531–539
  - CreateGeneralMessageL(), 537–538
    - creating a message, 535–536
    - dynamic menus, 533–534
    - getting started, 532–533
  - HandleSendAppUiCommandL(), 538–539
  - handling commands, 534–535
- CStreamStore, 158
- CSV (comma-separated value), 77, 707

CSY (serial communications module), 432, 434–435, 439, 708  
 CTypefaceStore, 571  
 Custom controls:  
   creating in a dialog class, 301–302  
   in dialogs, 300–302  
 Custom notes, 311  
**CustomCtrlDlg** example application, 291, 300–302  
 Customized notes, 313–315  
   constructing/executing, 315  
 CWaitNoteContainer, 317  
 CWAPBoundCLPushService, 507–508  
 CWAPBoundDatagramService, 506–507  
 CWAPFullySpecCLPushService, 507–508  
 CWAPFullySpecDatagramService, 507  
 CWindowGc, 563  
 CWsBitmap, 578  
 CWsScreenDevice, 563

## D

Dangling pointers, 244  
 data folder, 38–39  
 Data queries, 328–330  
   constructing/executing, 329–330  
   defining in resource, 328–329  
   types of, 326  
 DataAddress(), 581  
 Datagrams, 439–440, 457, 507, 708  
**DataQuery** example application, 327–330  
 Date editor, 425  
 Daytime, 431, 460, 463, 708  
 Debugging, 658, 686–696  
   applications on target, 692–696  
     building the application, 695  
     closing a session, 696  
     debugging on a device, 695–696  
     GDB initialization file, creating, 694–695  
     installing and configuring gdbstub, 693–694  
     installing the application on a device, 695  
     setting up the serial link, 693  
   applications on the emulator, 687–689  
   defined, 686  
   emulator lockup, 690–691  
   macros, 689  
   memory leaks, finding, 691–692  
   resource files, 689–690  
   screen flicker, removing, 690  
 Decoding, 595–597  
   Series 60 1.x, 595–596  
   Series 60 1.x vs. Series 60 2.x, 603  
   Series 60 2.x, 596–597  
 def files, 72  
 Defensive programming, 661–664  
   assertions, 661–662  
   class variants, 662–663  
   heap testing, 664  
   Delete(), 101  
   Delete Field, 304–305, 306  
   Descent (metric), 571  
   Descriptor arrays, 125–126  
   Descriptors, 76–77, 97–109, 708  
     as arguments and return types, 106–108  
     classes, 99  
     defined, 97  
     hierarchy, 98–99  
     literals, 101–102  
     modifiable API, 100–101  
     nonmodifiable API, 99–100  
     package, 108–109  
     using, 102–106  
       HBufC, 102–105  
       TBuf, 105–106  
       TBufC, 105–106  
   Development process overview, 4, 6–9  
   Development reference, 25–73  
   multi-bitmaps:  
     bmconv.exe, 58–60  
     colors and palette support, 59–60  
   Development tools, 27  
   devices command, 29, 36  
 Dialog class, custom control, creating in, 301–302  
 Dialog-based architecture, 188–191  
   when to use, 199–200  
 Dialogs, 289–339, 708  
   common characteristics of, 291  
   constructing/executing, 297–298  
   custom controls in, 300–302  
   defined, 289  
   defining a menu for, 300  
   dialog class, writing, 294–295  
   dialog data, saving and validating, 295–296  
   forms, 290, 302–310  
   list, 290, 335–338, *See also* Selection list dialogs  
     defined, 335  
     markable, 338  
   modal, 291  
   modeless, 291  
   multipage, 298–300  
   nonwaiting, 291  
   notes, 290, 310–325  
   queries, 290, 326–335  
   resource, defining, 292–294  
   selection list:  
     adding icons to, 337–338  
     constructing, 337  
     defining in resource, 336–337  
     executing, 338  
   simple, creating, 292–298  
   standard, 290, 291–302  
     creating a simple dialog, 291–302  
     initializing dynamically, 296–297  
     waiting, 291  
   Dial-up networking (DUN), 472, 708  
 Digia Quality Kit, 673  
 Direct screen access (DSA), 588–594, 708  
   architecture, 589–590

CDirectScreenAccess, 589, 591–592  
   implementation, 593–594  
   key classes for, 590–593  
   MAbortDirectScreenAccess, 589, 592–593  
   MDirectScreenAccess, 589, 592–593  
   RDirectScreenAccess, 589, 590–591  
 DisplayMode(), 581  
 DLL (Dynamic Link Library), 19, 35–36, 38, 40,  
   70–72, 128, 708  
   polymorphic, 179, 708  
 DoCancel(), 130, 142  
 Document embedding, 43, 708  
 DoExample() function, 91  
 Domain name server (DNS), 441, 708  
 DOUBLE, 254  
 Double buffering, 585–586, 708  
 DrawArc(), 576  
 DrawDeferred(), 243, 567  
 DrawEllipse(), 574–576  
 DrawNow(), 243, 567–568  
 DrawPie(), 576  
 DrawPolygon(), 576–577  
 DrawRect(), 575  
 DrawTextVertical() function, 574  
 DSA, *See* Direct screen access (DSA)  
 DTMF (Dual Tone Multi-Frequency), 609, 708  
 Duration editor, 425  
 Dynamic menus, 261  
 Dynamically allocated memory, 33–34  
 DynInitMenuPanel(), 261

## E

EAikCmdExit, 259–260  
 EAknCmdExit, 259  
 EAknConfirmationNoteFlags flag, 314  
 EAknEditorFlagNoEditIndicators, 299  
 EAknSoftKeyBack, 260  
 EColor64K, 567  
 ECom, 174–175, 206–213, 708  
   conceptual overview, 208  
   DLL, 211–213  
   interface, 208–211  
**EComExample** example application, 175, 208  
 EDataTypePriorityLow, 45  
 EDataTypePriorityNormal, 45  
 Edit Label, 304, 306  
 Editors, 63, 296, 306, 393–427, 708  
   multi-field numeric (MFNEs), 397, 423–427  
   numeric, 397, 417–422  
   secret, 397, 422–423  
   styles, 416–417  
   text, 396, 397–417  
 EEikDialogFlagCbaButtons, 293  
 EEikDialog-FlagFillAppClientRect flag, 298  
 EEikDialogFlagNoDrag, 293  
 EEikDialogFlagWait, 293

EEventKey, 241  
 EEventKeyDown, 241  
 EEventKeyUp, 241  
 EFileRead, 149  
 EFileWrite, 149  
 Eikon, 177, 709  
 Elements application, 77, 80  
**Elements** example application, 77, 80, 92, 95, 103,  
   111, 118, 122, 132, 200  
 Ellipses, 575  
 EMCC Software Ltd., 8  
 Emulator, defined, 709, *See also* Series 60  
   emulators  
 Encoding, 597–601  
   Series 60 1.x, 598–599  
   Series 60 2.x, 599–601  
 Engine, 35, 81, 174, 197, 202, 709  
 Enumerated text setting item reference, 388  
 EnvironmentSwitch tool, 29, 62  
 epoc, 15, 17–18, 36, 46–47, 709  
 EpocCheck tool, 88, 671  
 EPOCHSIZE statement, 34  
 epoc.ini, 65  
 EPOCROC, 66  
 EPOCSwitch tool, 28, 62  
 EPOCToolbar, 62  
 EPOSTACKSIZE statement, 34  
 EPriorityStandard, 130  
 Error note, 311  
 ESimpleMenuCmdNewGame, 257  
 ESOCK, 439, 459, 709  
 ETel, 543–544, 546, 548–549, 709  
 Event handling, 224, 239–244  
   key events, 239–242  
     control stack, 241  
     focus, 240–241  
     offering, 241–242  
     observers, 244  
     redraw events, 243  
 Events, 239–242, 709  
   offering, 241–242  
   redraw, 243  
 EVerticalHatchBrush, 567  
 Exception handling, 76  
 Exceptions, 84–88, 709  
 Execution, Series 60 1.x vs. Series 60 2.x, 603  
 EXPORTUNFROZEN keyword, 72  
 Externalization, 550, 798  
 ExternalizeL(), 154–156, 581

## F

Factory function, 153, 157, 709  
 FBSCli library, 559  
 FFS (Flash filing system), 681, 709  
 Fields, initializing, 250  
 File handling, 200–201

File logging, 675–677  
 File Server, 132, 143–148, 165–166, 170, 175, 177, 433, 568, 709  
 File system, and good application behavior, 219  
 Files, 143–151, 709  
   RFile API, 147–151  
   RFs API, 144–147  
   Symbian OS filenames and pathnames, 143–144  
 FilmReel example application, 619, 628, 630  
 FilmReel2 example application, 619, 628  
 Find(), 100  
 Fixed-period progress notes, 324–325  
 Flush, 160, 690, 709  
 FocusChanged(), 240  
 Fold, 709  
 Font and Bitmap Server, 558  
 Fonts and text, 554, 569–574  
   key classes/functions, 571  
   key font classes, using to enumerate through all available fonts, 572–573  
   measurements, 570–571  
   text effects, 573–574  
**FontsAndText** example application, 555, 572–573  
 Format(), 101  
 Forms, 290, 302–310, 710  
   Add Field, 304, 306  
   creating in an application, 306–310  
   defined, 302  
   defining in a resource, 306–308  
   Delete Field, 304, 306  
   Edit Label, 304, 306  
   edit mode, 302–303  
   editing, 305  
   executing, 309–310  
   form lines, 303–304  
   form-derived class, creating, 308–309  
   soft keys, 304–306  
   view mode, 302–303  
 Framework, 36–37, 710  
 Front-end-processor (FEP), 408, 709  
 FsSession(), 144, 568  
 FTP (File Transfer Protocol), 448, 457, 710  
 Functional testing, 667–668

## G

gcc, 70  
 GDI (Graphics Device Interface), 559, 710  
 GDI library, 559  
 GET request, 710  
   making, 495–496  
 GetDir(), fetching a directory listing using, 146–147  
 GetPixel(), 581  
 GIF (Graphics Interchange Format), 559, 710  
 Global editor, 397

Global message queries:  
   creating, 333–334  
   handling dismissal of, 334–335  
 Global notes, 311, 325  
 Global queries, 327, 332–335  
   complexity of, 332–333  
   defined, 332  
   types of, 335  
**GlobalQuery** example application, 327, 333–335  
 GMS grid, 367  
 GNU C++ compiler (gcc), 9, 671, 710  
 GNU, defined, 710  
 Good application behavior, 174, 217–220  
   Active Objects and responsiveness, 220  
   adopting a skeptical/critical approach to development, 217  
   checking disk space before saving data, 219  
   exiting gracefully, 218–219  
   and file system, 219  
   handling Window Server-generated events, 218  
   hard-coding and magic numbers, 220  
   and system watchdogs, 219  
   timers, 220  
 GPRS (General Packet Radio Service), 66  
 Granularity, 111, 117, 710  
 Graphical User Interface (GUI), 4–5, 77, 710  
   application build process summary, 36  
 Graphics device classes, 563  
 Graphics devices and graphics contexts, 562–565  
 Graphics libraries, 559  
 Grids, 16–17, 342–343, 364  
   basics, 368–375  
   concrete grid class, creating, 372–375  
   defined, 364, 710  
   defining using resources, 368–371  
   GMS grid, 367  
   markable, 364, 375–376  
   menu, 364  
   monthly calendar grid, 365–366  
   multiselection, 364  
   orientation, 365  
   pin-up board grid, 366  
   selection, 364  
   types of, 364  
   using, 367–376  
 group folder, 38–39  
 GSM (Global System for Mobile communication), 502, 651, 710

## H

h, 9  
 HalView example application, 618, 619  
 Handle, 129, 131–132, 710  
 HandleCommandL(), 259  
 HandleControlEventL(), 244



- HandleResourceChange(), 239
  - Hardware Abstraction Layer (HAL), 618, 649–651, 710
    - phone IMEI number, 651
  - HBufC, 95, 99, 102–105, 106
  - Header part, resource files, 247–248
    - application information resource, 248
    - document name buffer, 248
    - include statements, 247
    - NAME statement, 247
    - RSS\_SIGNATURE, 248
  - Heap, 33–34, 78, 710
    - allocation, 87
    - defined, 78
  - Heap testing, 664
  - Height (metric), 571
  - HelloWorld GUI application, 30–46
    - aif files, 43–46
      - aifbuilder tool, 43
      - aiftool utility, 43
      - captions, 45–46
      - file localization, 44
      - icons, 44
      - MIME support, 44–45
    - application resource files, 40
    - avkon.hrh, 41
    - bld.inf, 6–8, 13, 30
    - executable and runtime files, 36–38
    - HelloWorldCon, 46–47
      - building, 47–49
      - emulator executable files, 49
      - mmp, 48–49
      - running, 47–49
      - target executable files, 49–50
    - localization of applications and resources, 42–43
    - mmp, 31–33
    - project files and locations, 38–39
    - resource compiler, 41–42
    - source files, 39–41
    - stack and heap sizes, 33–34
    - UIDs (unique identifiers), 31–33
  - HelloWorld project, 4–5
  - HelloWorld\_caption.rsc file, 37, 40, 45–46
  - HelloWorld.aif file, 37
  - HelloWorld.app file, 36–37
  - HelloWorldApp.cpp, 39
  - HelloWorldApp.h, 39
  - HelloWorldAppUi.cpp, 39
  - HelloWorldAppUi.h, 39
  - HelloWorld.cbx, 7
  - HelloWorldCon, 46–47
    - building, 47–49
    - emulator executable files, 49
    - mmp, 48–49
    - running, 47–49
    - target executable files, 49–50
  - HelloWorldCon** example application, 46–48, 50, 55
  - HelloWorldContainer.cpp, 39
  - HelloWorldContainer.h, 39
  - HelloWorldDocument.cpp, 39
  - HelloWorldDocument.h, 39
  - HelloWorld.dsp, 7
  - HelloWorld.dsw, 7
  - HelloWorldLoc, 27, 53
    - .pkg file, 56
  - helloworld.mmp, 6, 7, 12, 13
  - HelloWorld.rsc, 36–37
  - Host, 497
  - HTML (HyperText Markup Language), 491, 710
  - HTTP (HyperText Transfer Protocol), 490, 491–501, 711
    - data suppliers, 493
    - filters, 493
    - headers, 492
    - HTTPExample** application, 493–501
      - sessions, 491–492
      - transactions, 492
    - HTTPExample** application, 490, 493–501
      - GET request, making, 495–496
      - implementation, 493–494
      - opening a session, 494–495
      - POST request, making, 500–501
      - request headers, adding, 496–497
      - submitting the request/receiving the response, 497–500
      - terminating the session, 501
    - HTTPExample** example application, 493
  - HyperTerminal, 675, 683–684
- 
- ## I
- 
- IANA (Internet Assigned Numbers Authority), 460, 711
  - IAS (IrDA Information Access Service), 466, 711
  - ICMP (Internet Control Message Protocol), 443, 457–458, 711
  - ICO, 559, 711
  - IDE, *See* Integrated Development Environment (IDE)
  - Image conversion, 594–601
  - Image manipulation, 555, 594–604
  - Image rotation, 601–602
    - Series 60 I.x, 601
    - Series 60 I.x vs. Series 60 2.x, 603
    - Series 60 2.x, 601–602
  - Image scaling, 602–604
    - Series 60 I.x, 602
    - Series 60 I.x vs. Series 60 2.x, 603
    - Series 60 2.x, 602–603
  - imagefile, 317
  - imageid, 317
  - ImageManip** example application, 555, 594
  - imagemask, 317
  - IMAP4 (Internet Message Access Protocol, v.4), 516

IMEI (International Mobile Station Equipment Identification), 651, 711  
 Import library, Series 60 1.x vs. Series 60 2.x, 603  
**Import Mobile** example application, 13  
 Inbox, watching, 540–543  
 inc folder, 38  
 Include header, Series 60 1.x vs. Series 60 2.x, 603  
 Incoming messages, 539–543  
   APIs, 539–540  
   watching the inbox, 540–543  
 Indicators, 284–286  
 Information note, 311  
 Insert(), 101  
 install folder, 38–39  
 Installation File Generator, 50–52, 711  
 Integrated Development Environment (IDE), 4,  
   7–9, 12–15, 19, 28–29, 40, 47–48, 710  
   command-line tools vs., 8–9  
   installation tips for, 27  
 Integration testing, 666–667  
 Interfaces, 40–41, 46, 80, 82, 111, 118  
   defined, 711  
 Internalize, 156–157, 711  
 InternalizeL(), 156–157, 581  
 Internationalization, 174, 213–217, 221, 711  
   general guidelines for developers, 213–216  
   localization, OS support for, 216–217  
 Internet access point (IAP), 459–461, 711  
 Internet Protocol (IP), 424, 443, 457, 504, 712  
 Inter-Process Communication (IPC), 170, 458, 711  
 IP address, 390, 424, 426, 459, 462, 506, 712  
 IP address editor, 424  
 IP address editor setting item reference, 390  
 IR (infrared), 443, 469, 675, 712  
 IrCOMM (Infrared Communications), 434, 466, 467,  
   712  
 IrDA (Infra Red Data Association), 66, 465, 467, 712  
 IrDA Serial API, 469  
 IrDA Sockets API, 467–468  
 IrLAN (Infrared Local Area Network), 467, 712  
 IrLAP, 466  
 IrLMP, 466, 712  
 IrMUX, 443, 467–468, 712  
 IrOBEX (Infrared Object EXchange), 467, 470, 712  
 IrSerial, 431  
**IrSerial** example application, 432, 434–435, 469  
 IrSockets, 431  
**IrSockets** example application, 439–440, 444–445,  
   448  
 IrTranP, 467, 469–470, 712  
 IsFocused(), 240  
 ISP (Internet Service Provider), 66, 712

---

## J

Java, 500, 557, 712  
 Jeteye ESI-9680 (Extended Systems, Inc.), 66

JPEG (Joint Photographic Experts Group), 59,  
 594–595, 712  
 Just-in-time (JIT), 712  
 debugging, 689

---

## K

KAknsMessageSkinChange, 239  
 KEikMessageColorSchemeChange |, 565  
 Kernel, 127, 131, 166–167, 169, 712  
 KERN-EXEC 3, 34  
 KErrNoMemory, 87  
**Key events**, 239–242  
   control stack, 241  
   focus, 240–241  
   offering, 241–242  
 KeyMap keyword, 65  
 KUidMsgTypeBt, 516  
 KUidMsgTypeIMAP4, 516  
 KUidMsgTypeIr, 516  
 KUidMsgTypeMultimedia, 516  
 KUidMsgTypePOP3, 516  
 KUidMsgTypeSMS, 516  
 KUidMsgTypeSMTP, 516

---

## L

L2CAP (Bluetooth Logical Link Control and  
 Adaptation Protocol), 471, 712  
 LANG, 32  
 Last calling number, retrieving, 550–551  
 LDRA Testbed, 672  
 Leave, 84–88, 712  
 Leavescan, 671  
 Leavescan tool, 88  
 Leaving, 84  
 Leaving issues, and the cleanup stack, 88–91  
 Left(), 100  
 Left adjust (metric), 571  
 Length(), 99  
 LIBRARY statement, 32–34  
 LINK, 254  
 Linked list, 117, 713  
 List dialogs, 335–338, *See also* Selection list  
   dialogs  
     defined, 335  
     markable, 338  
 List queries, 327, 330–332  
   creating/executing, 331–332  
   multiselection, 332  
   resources, 330–331  
 Listen queue, 445  
 Lists, 341–393  
   basics, 342, 343

defined, 342  
 grids, 342–343  
 settings, 343  
 vertical, 342–364  
 Literal search, 66  
 Literals, 101–102  
 LLINK, 254  
 LoadFromCsvFileL(), 87  
 Locale, 42, 53, 56, 214, 216–217, 296, 713  
 Localization of applications, 26  
 Locally scoped variables, 33  
 Log engine, 625–627, 713  
   log entries, reading, 625–627  
 Logical driver, 432  
 LONG, 254  
 LTEXT, 254

**M**

MAbortDirectScreenAccess, 589, 592–593  
 Machine UIDs, 69, 713  
 Macros:  
   debugging, 689  
   defined, 713  
 Magic numbers, 220, 713  
 Main pane, 11, 286  
 makefile, 14, 713  
 makekeys.exe, 50, 51–52  
 makesis.exe, 50, 51–52  
 Markable grids, 364, 375–376  
   creating, 364  
   drawing, 365  
   icons, setting, 364–365  
 Markable list dialogs, 338  
 Markable lists, 344, 346, 357–360  
   defining using resources, 358  
   dynamically changing marked list items, 359–360  
   marking commands, handling, 358–359  
**MarkableList** example application, 344, 347,  
   357–359, 376  
 Mask, 60, 315, 713  
 Masking color, 60  
 MaxLength(), 101  
 MaxSize(), 101  
 MBM (multi bitmap file), 59, 274, 279, 349–350,  
   559, 713  
 MBMViewer, 62  
**M**-classes, 80  
 MCoeControlObserver, 244  
 MCsvFileLoaderObserver interface, 80–81  
 MDirectScreenAccess, 589, 592–593  
 MEDIACLIENTIMAGE library, 559  
 Memory allocation, 33–34  
 Memory leak, 89, 483, 573, 691–692, 713  
 Menu grids, 364  
 Menu lists, 344, 345  
 MENU\_BAR, 256–258

MENU\_ITEM, 257–258  
 MENU\_PANE, 257–258  
 MENU\_TITLE, 257–258  
 Menus, 224, 254–264, 713  
   basics, 256–260  
   context-sensitive, 262–264  
     defining using resources, 262  
     displaying, 262–264  
   defined, 254  
   defining using resources, 256–258  
   dynamic, 261  
   menu commands, handling, 258–260  
   submenus, 255  
   using in a control, 258  
 Message queries, 327  
 Messaging, 490, 508–543, 713  
   APIs, 512–514  
     Client MTM API, 512–526  
     CSendAppUi API, 513–514  
     Send-As API, 513  
   architecture, 509  
   entries, 509  
   entry storage, 510  
   file system, 510  
   generic entry handling/unified inbox, 509–510  
   incoming, 539–543  
     APIs, 539–540  
     watching the inbox, 540–543  
   key classes/data types, 511–512  
     CMsvEntry, 511  
     CMsvEntrySelection, 512  
     CMsvOperation, 512  
     CMsvSession, 511  
     CMsvStore, 511  
     TMsvEntry, 511  
     TMsvID, 511  
   key concepts, 508–510  
   Message Index, 510  
   messaging server and session, 508  
   Messaging Store, 510  
   MTMs (message type modules), 509  
**Messaging** example application, 22  
 Messaging view switching, 623  
 Metrowerks CodeTEST, 672  
 Metrowerks CodeWarrior, 6, 8  
 Metrowerks CodeWarrior C++ IDE, 9  
 Metrowerks CodeWarrior Development Studio for  
   Symbian OS v.2.5, 671  
 Metrowerks CodeWarrior IDE, use of devices with,  
   29  
 MFNE (Multi-Field Numeric Editor), 423–427  
   defined, 714  
 MFNEs, *See* Multi-field numeric editors (MFNEs)  
 MGraphicsDeviceMap, 571  
 Microsoft Visual C++:  
   Symbian OS variable expansion, 63  
   syntax highlighting, 63  
 Microsoft Visual C++ 6.0, 6  
   and .NET, 671  
 Microsoft Visual C++ IDE, 9

Microsoft Visual Studio .NET, 64  
 Mid(), 100  
 .midi format, 604  
 MIME, defined, 713  
 MIME support, .aif files, 44–45  
 Mixins, 80, 176, 259, 261, 483, 496, 527, 592, 630, 713  
 Mkdir(), 145  
 MkdirAll(), 145  
 MMdaImageUtilObserver class, 594  
 .mmp file, 6–8, 12, 46, 72, 251, 649  
 MmpClick, 63  
 MMS MTM:  
   adding attachments, 524–526  
   creating the message, 523–524  
   sending an entry, 526  
   using, 523–526  
   validating an entry, 526  
 MMS (multimedia messaging service), 401, 490, 508–509, 513–514, 516, 523–524, 537  
   defined, 714  
 Mobile Innovation TRY, 673  
 MObjectProvider, 239  
 Modal, 142, 189, 291–292, 295, 297, 714  
 Modal dialogs, 291  
 Modeless, defined, 189–191, 291, 714  
 Modeless dialogs, 291  
 Monthly calendar grid, 365–366  
 Move (metric), 571  
 MSDN (Microsoft Developer Network), 19, 714  
**MsgObserver** example application, 491, 539–540  
 MTM (Message Type Module), 512–513  
   defined, 714  
**MtmsExample** example application, 490, 514  
 Multi Media Server, 558–559  
 Multi-field numeric editors (MFNEs), 397, 423–427  
   date editor, 425  
   defining in a resource, 426  
   duration editor, 425  
   getting/setting a value of, 427  
   instantiating, 426  
   IP address editor, 424  
   number editor, 424  
   range editor, 424  
   time and date editor, 425  
   time editor, 425  
   time offset editor, 425  
   using an MFNE, 425–427  
 Multihoming, 459–467  
   closing connections, 463–464  
   explicit connections, 460–463  
   implicit connections, 464–465  
   RConnection API, 459–460  
   transferring data, 463  
 Multimedia, 553–615  
   animation, 554–555, 582–615  
   audio, 555, 604–614  
     audio data, 610–612  
     audio device modes, 607–608

    .midi format, 604  
     recording, 605–609  
     Series 60 audio libraries, 604  
     streaming, 612–614  
     tones, 609–610  
     .wav format, 604  
   basic drawing, 554, 559–569  
   bitmaps, 554, 578–581  
   fonts and text, 554, 569–574  
   image manipulation, 555  
   Series 60 graphics architecture, 554, 556–559  
   shapes, 554, 574–577  
 Multimedia Messaging Service (MMS), *See also*  
   MMS MTM  
   defined, 523  
**MultiMediaF** example application, 555, 594  
 Multipage dialogs, 298–300  
 Multiple SDKs, using, 6  
 Multiselection grids, 364  
 Multiselection list queries, 332  
 Multiselection lists, 344, 346  
 Multitasking, 77, 127–128, 142, 714  
 Multithreading, 127, 594, 600, 603

## N

NAME\_OF\_APP\_caption.rss, 45–46  
 Namespaces, 81, 113–114  
   defined, 714  
 Naming conventions, 76, 77–81  
   **C**-classes, 79  
   **M**-classes, 80  
   namespaces, 81  
   **R**-classes, 79–80  
   **S**-classes, 79  
   **T**-classes, 78–79  
 Navigation pane, 265–266, 274–286  
   displaying a label in, using resources, 281–283  
   displaying an image in, using resources,  
     283–286  
   displaying tabs in, 274–276  
   using resources, 278–281  
   indicators, 284–286  
   main purpose of, 274  
**NavigationPane** example application, 225,  
   278–281, 283  
 NBitmapMethods, 81  
 NEikonEnvironment, 81  
**New Symbian GUI** example application, 61  
 NewL(), overloading to take a read stream, 157  
 NewLC(), 94, 325  
   overloading to take a read stream, 157  
 NIFMAN (Network Interface Manager), 458, 714  
 Nokia Testing Suite, 673  
 Non-skin-aware controls, 239  
 Nonwaiting dialogs, 291

Non-Window-owning controls, 228–229  
**NormalFont** (), 568  
**Notes**, 290, 310–325  
   customized, 313–315  
   constructing/executing, 315  
   defined, 714  
   global, 325  
   predefined, 311–312  
   progress, 319–325  
   completion of, and user cancellation, 323–324  
   creating for a variable-length process, 321  
   declaring, 320–321  
   defined, 319–320  
   executing, 321–322  
   fixed-period, 324–325  
   updating as a process executes, 322–323  
 wait, 316–319  
   defining in resource, 316–317  
   **MAknBackgroundProcess**-derived class, 318–319  
   wrapper object, constructing and executing, 316–318  
   wrapped, 310–313  
**Notification colors**, 60  
**Number editor**, 424  
**Numeric editors**, 397, 417–422  
   characteristics of, 418  
   configuring, 418–422  
   defined, 417  
   fixed-point editor resource, defining, 419–420  
   integer editor resource, defining, 419  
   obtaining/validating values in, 420–421  
   resources/classes, 418  
   setting values in, 421–422  
   types of, 417  
**NumericEditor** example application, 397, 418–421, 427

## O

**OBEX** (Object Exchange), 22, 470, 714  
**Observer Design Pattern**, 80  
**Observers**, 244  
**Offering events**, 241–242  
**OfferKeyEventL** (), 241, 262  
**Offscreen bitmaps**, 585–586  
**OOM** (out of memory), 664, 685–686, 714  
**Open** (), 147–148  
**Operating system (OS)**, 15, 34, 126, 477  
   defined, 714  
**operator** (), 102  
**operator[]** (), 100  
**operator<** (), 100  
**operator=** (), 100  
**operator!=** (), 100  
**operator==** (), 100

**operator>** (), 100  
**OpponentForm** example application, 305–310, 419  
 Optionally skin-providing controls, 238  
**Options menu**, 11, 183, 255–256, 262, 266,  
   269–270, 273, 304–306, 309, 313, 316, 320, 337,  
   343, 346, 355–359, 378, 383, 714  
**Out-of-memory testing**, 684–686

## P

**Package descriptors**, 108–109  
**Packet Switched Data (PSD)**, 459, 714  
**Panes**, 224, 264–286  
   context pane, 272–274  
   main pane, 286  
   navigation pane, 274–286  
   soft key pane, 286  
   status pane, 264–268  
   title pane, 268–272  
**Panics**, 34, 64, 84–88, 91, 100–101, 105, 107, 113,  
   125–139, 142–143, 152, 248, 294, 423–424, 714  
   in Series 60 emulator, 67  
**Password editor setting item reference**, 392  
**Pathnames**, Symbian OS, 143–144  
**PC-based development options**, 9  
**PC-based platform emulators**, 6  
**PC-Lint**, 671  
**PCM** (Pulse Code Modulation), 612, 715  
**PDP** (Packet Data Protocol), 459, 715  
**Pens and brushes**, 567  
**Performance testing**, 668  
**Petran**, 20, 71  
**Phonebook engine**, 632–638  
   creating and accessing phonebook items, 632–635  
   phonebook searching, 636–637  
   receiving notification, 635–636  
   speed dialing and common dialogs, 637–638  
**Phonebook view switching**, 620–621  
**Photo album engine**, 646–649  
   photo album folders, accessing, 646–648  
   photo album UI components, 648–649  
**Photo Album view switching**, 622–623  
**Physical driver**, 432  
**Pies**, 576  
**Pin-up board grid**, 366  
**pkg** file format, 52–57  
   conditional component installation, 57  
   multicomponent installation, 57  
   multi-locale installation, 55–56  
   running executables during installation, 55  
**Plain editor**, 397  
**PlainTextEditor** example application, 397, 404–406, 410, 427  
**Platform UID** (platform identification code), 53, 715

PlaySelectedGame(), 262  
 Plug-in, 13, 165, 206–207, 715  
 PNG (Portable Network Graphics), 559, 715  
 Polygons, 576–577  
 POP3 (Post Office Protocol, v. 3), 66, 512–513, 516, 715  
 PopAndDestroy(), 91  
 Pop-up menu lists, 361–364  
   creating, 361–362  
   displaying and handling selections, 363–364  
   list classes/item definitions, 362–363  
   setting the title for, 362–363  
 PopUpList example application, 344, 347–348, 361–363  
 Port number, 444–445, 447, 458, 469, 481, 484–485, 504, 715  
 POST request:  
   defined, 715  
   making, 500–501  
 Predefined notes, 311–312  
 Profiles view switching, 623  
 Profiling, 670  
 Program stack, defined, 78  
 Progress notes, 311, 319–325  
   completion of, and user cancellation, 323–324  
   creating for a variable-length process, 321  
   declaring, 320–321  
   defined, 319–320  
   executing, 321–322  
   fixed-period, 324–325  
   updating as a process executes, 322–323  
 Project|Properties menu item, 58  
 projectname.mmp, 6–7  
 PRT, 500–501, 504  
 Ptr(), 99

## Q

Quality assurance (QA), 658, 659–664  
   coding standards, 659–661  
   defensive programming, 661–664  
     assertions, 661–662  
     class variants, 662–663  
     heap testing, 664  
   defined, 715  
 Queries, 290, 298, 304–305, 309, 326–335  
   confirmation, 326  
   data, 328–330  
     constructing/executing, 329–330  
     defining in resource, 328–329  
     types of, 326  
   defined, 210, 326, 715  
   global, 327, 332–335  
     complexity of, 332–333  
     defined, 332  
     types of, 335

global message queries:  
   creating, 333–334  
   handling dismissal of, 334–335  
   list, 327, 330–332  
     creating/executing, 331–332  
     multiselection, 332  
     resources, 330–331  
   message, 327

## R

RAM drive, 326, 715  
 Range editor, 424  
**RArray** and **RPointerArray** types, 110–117  
   basic APIs, 110–113  
   finding, 115–117  
   sorting, 113–115  
 RAS (Remote Access Service), 326, 715  
**R**-classes, 79–80, 96–97  
 RDirectScreenAccess, 589, 590–591  
 Read(), 148–149  
 ReadResource(), 568  
 Recording, 605–609  
 Recovery testing, 669  
 Rectangles, 575  
 Redraw events, 243  
 ReportEventL(), 244  
 RESOURCE, 32  
 Resource files, 224, 245–254  
   defined, 715  
   punctuation, 252  
   resource structures, creating, 253–254  
   string resources, 250–251  
   STRUCT field types, 254  
   structure, 247–252  
     body part, 247, 249–253  
     header part, 247–248  
   syntax, 245–247  
 Resource management, 76  
 RFC 2616, 491, 497–498, 715  
 RFCOMM, 444, 472, 476, 484, 486, 716  
**RFile** API, 147–151  
   opening and closing, 147–148  
   reading, 148–149  
   seeking, 151  
   writing, 149–151  
**RFs** API, 144–147  
   Close(), 144–145  
   Connect(), 144–145  
   GetDir(), fetching a directory listing using, 146–147  
 RGB, 60, 581, 716  
 Rich text editor, 398  
   character formatting, applying, 413  
   configuring, 410–416  
   constructing from a resource, 415–416

- control:
    - creating, 411–412
    - using, 416
  - cursor, positioning, 414–415
  - defining a resource, 410–411
  - formatting attributes, setting, 412
  - setting text in, 413
  - RichTextEditor** example application, 397, 410–413, 415–417, 427
  - `Right()`, 100
  - Right adjust (metric), 571
  - RISC (Reduced Instruction Set Computer), 678, 716
  - `Rmdir()`, 145
  - ROM (read-only memory), 37, 151, 680, 681, 716
  - `RReadStream`, 153–154
  - `.rss`, 9
  - RS-232 standard, 472, 675, 716
  - `RunDlgLD()`, 297–298
  - `RunError()`, 131
  - `RunL()`, 130–131
  - `RWriteStream`, 152–153
- S**
- `sample.app`, 694
  - `Save()`, 304, 581
  - Scheme colors, 260
  - S**-classes, 79
  - Screen coordinates and geometry, 562
  - `ScreenDevice()`, 568
  - Scripted testing, 672–673
  - SDK (Software Development Kit), 4–6, 10, 13–15, 17, 23, 26, 28–29, 31, 41, 45, 47, 50, 52, 59, 61–63, 66–67, 110, 201, 214, 339, 374, 407, 426, 676, 716
  - Secret editors, 397, 422–423
    - defining a resource, 422–423
    - instantiating, 423
    - resources/classes, 423
  - Secure sockets, 451–457
    - Series 60 1.x, 452–456
    - Series 60 2.x, 456–457
  - `Seek()`, 151
  - Selection button, 36
  - Selection grids, 364
  - Selection key, 255, 262–263, 294, 343, 345–346, 378, 383, 686, 716
  - Selection list dialogs, 336–338
    - adding icons to, 337–338
    - constructing, 337
    - defining in resource, 336–337
    - executing, 338
  - Selection lists, 344
  - Semaphore, 127–128, 135, 140–141, 169, 716
  - Send-As API, 513, 526–530
    - Send-As object, creating, 527
  - SendAsExample** example application, 527–528
  - Serial communication, 430, 431–438
    - Comms Server (C32):
      - connecting to, 434
      - starting, 434
    - CSY, loading, 434
    - defined, 430, 716
    - serial device drivers:
      - loading, 432–433
    - serial port:
      - closing, 438
      - configuring, 435–437
      - opening, 434–435
      - transferring data over, 437–438
      - using, 432–438
  - Serial Communications Server, 431
  - Series 60 1.x, differences between Series 60 2.x and, 603
  - Series 60 1.x WAP APIs, 503–506
    - connected WSP session, 504–505
    - connectionless WSP, 505–506
    - WAP datagrams, sending, 503–504
  - Series 60 2.x WAP APIs, 506–508
    - `CWAPBoundCLPushService`, 507–508
    - `CWAPBoundDatagramService`, 506–507
    - `CWAPFullySpecCLPushService`, 507–508
    - `CWAPFullySpecDatagramService`, 507
    - WAP datagrams, sending, 506
  - Series 60 application wizards, 60–62
    - Borland C++BuilderX Mobile, 61
    - Metrowerks CodeWarrior, 62
    - Microsoft Visual C++ 6.0, 61
  - Series 60 audio libraries, 604
  - Series 60 C++ software development kits (SDKs), 4, 5–6
  - Series 60 emulators, 4, 10–12
    - application panics in, 67
    - building, 4, 12–14
      - from the command line, 12–13
      - from an IDE, 13–14
      - using Borland C++BuilderX, 14
      - using Borland C++IDE Builder 6, 13–14
      - using CodeWarrior IDE, 14
      - using Microsoft Visual C++ IDE, 13
    - configuration, 64–66
      - options and information sources, 65–66
    - debug mode, 17
    - executable locations, 15–17
      - debug build emulator, 17
      - release build emulator, 15
    - hardware color capabilities of, 59–60
  - HelloWorld application:
    - debugging, 19
    - locating/running, 18
    - main pane, 11
    - Options menu, 11
    - running, 4, 15–19

- Series 60 emulators, *continued*
  - from Borland C++Builder 6 Mobile Edition IDE, 18
  - from Borland C++BuilderX IDE, 18
  - from the CodeWarrior IDE, 18
  - from a command prompt, 17
  - from the Visual C++ IDE, 18
- shortcut keys, 699–703
- Series 60 graphics architecture, 554, 556–559
  - Font and Bitmap Server, 558
  - graphics libraries, 559
  - Multi Media Server, 558–559
  - Window Server, 556–557
- Series 60 graphics libraries, 559
- Series 60 Software Development Kit (SDK), 4, 10
  - installation tips for, 27, 64–66
- Series 60-specific coding identification colors-, 60
- Series60Tools folder, 62
- Server, 53, 66, 79, 127, 165–170, 716
- Server push, 503
- Server sessions, 165–166
  - and Inter-Process Communication (IPC), 166–168
  - server review, 169
  - subsessions, 169–170
- Service Discovery Database (SDD), 472–473, 474, 716
  - closing, 477
- Service Discovery Protocol (SDP), 472, 477, 716
- Session, 716
- SessionPath(), 145
- SetActive(), 130
- SetContainerWindowL(), 567
- SetDimmed(), 261
- SetFocus(), 240–241
- SetLength(), 101
- SetMax(), 101
- SetObserver(), 244
- SetObserver(NULL), 244
- SetSessionPath(), 145
- Settings lists, 342, 343, 377–379, 381, 383, 385, 716
  - basics, 379–392
    - binary switch setting item reference, 390
    - constructing, 383–384
    - defining using resources, 379–381
    - deriving, 381–382
    - enumerated text setting item reference, 388
    - IP address editor setting item reference, 390
    - password editor setting item reference, 392
    - setting items, 377
      - creating, 382
      - types of, 384
    - settings list values, changing, 383
    - slider setting item reference, 384–385
    - text editor setting item reference, 387
    - time or date editor setting item reference, 389
    - using, 378–392
    - volume setting item reference, 386
- SettingsList example application, 378–384, 391
- ShapeDrawer example application, 175, 179, 191, 195, 196, 200, 219
- Shapes, 553, 554, 574–577
  - arcs, 576
  - ellipses, 575
  - pies, 576
  - polygons, 576–577
  - rectangles, 575
- Shapes example application, 554, 555, 575, 577
- Shortcut keys:
  - Series 60 emulators, 699–703
  - drawing, 700
  - hardware emulation, 702
  - miscellaneous, 703
  - resource allocation, 701
  - Window Server, 700–701
- ShowNoteL(), 325
- Siemens SX1 smartphone, 6
- Simple controls, 225–228
  - creating, 229–232
- Simple dialogs, creating, 292–298
- SimpleDlg example application, 175, 190, 291–294, 296–298, 300
- SimpleList example application, 344, 347–352, 354, 358, 368, 372
- SimpleMenu example application, 224, 256, 259
- SimulateKeyEventL(), 568
- .sis file, 50
  - building, 21–22, 58
  - defined, 716
  - installing, 22
- sisar utility, 51–52, 57, 717
- Size(), 99
- Skiing example application, 555, 588, 590–591, 593
- Skin-aware controls, defining, 239
- Skin-observing controls, 238
- Skins, 224, 237–239
  - compulsory skin-providing controls, 238
  - defined, 237, 717
  - non-skin-aware controls, 239
  - optionally skin-providing controls, 238
  - skin-aware controls, defining, 239
  - skin-observing controls, 238
- Slider setting item reference, 384–385
- SMIL (Synchronized Multimedia Integration Language), 523–526, 717
- SMS Inbox, 63
- SMS MTM:
  - creating an entry, 518–520
  - sending an entry, 520–523
  - in Series 60, 517–518
  - using, 517–523
  - validating an entry, 520
- SMS OTA (Over the Air), 559
- SMS (Short Message Service), 63, 255, 490, 502, 504, 508, 514, 517–520, 625, 633, 638, 668, 717
- SMTP (Standard Mail Transport Protocol), 457, 717



- Socket communication, Bluetooth (BT), 484–486
- Sockets, 84, 430, 438–457, 717
  - address family constants, 443
  - client socket, connecting to, 446–447
  - client/server, 439
  - closing, 450–451
  - connected sockets, 440–451
  - connecting to the Symbian OS Socket Server, 441–442
  - connectionless and connected sockets, 439–440
  - opening, 442–444
  - protocol constants, 443–444
  - secure, 451–457
  - on Series 60, 438–439
  - server socket, connecting to, 444–446
  - transferring data over, 448–450
  - type constants, 443
- Soft key, 10–11, 183, 193, 255–256, 259–260, 304–306, 311, 312, 314, 328, 335, 364, 378, 387
- Soft key pane, 286
- Software Development Kits (SDKs):
  - installation tips for, 27
  - versions/selection, 27–29
    - Series 60 Version 1.x SDKs, 28–29
    - Series 60 Version 2.x SDKs, 29
- SOURCE, 32
- SOURCEPATH, 32
- Special character tables, 403
- Sprites, 58, 717
- src folder, 38
- SRLINK, 254
- SSL handshake, 453
- SSL (Secure Socket Layer), 451, 453, 456, 487, 717
- Stack:
  - Bluetooth (BT), 471
  - defined, 78, 717
- Standard application views:
  - using, 619–624
  - using resources, 618
- Standard dialogs, 290, 291–302
  - initializing dynamically, 296–297
  - simple dialog, creating, 291–302
- Standard Template Library (STL), 76
- Start(), 130
- StartVibra(), 653–654
- Static colors, 60
- Status pane, 10, 264–268
  - basics, 266–268
  - changing the visibility of, 266–267
  - context pane, 265
  - defined, 264
  - navigation pane, 265–266
  - position of, 264
  - size, handing a change in, 268
  - subpanes, 265
  - title pane, 265
- StatusPane example application, 225, 266–268
- STL (Standard Template Library), 76, 109, 717
- StopVibra(), 653
- Stores, 77, 157–164, 717
  - CDirectFileStore, 159–160
  - defined, 157, 717
  - reading from, 162–163
  - steps in using, 163–164
  - Stream Dictionary, 161–162
  - types of, 158
- Stream Dictionary, 161–162
- Streaming, 612–614
- Streams, 77, 151–157
  - defined, 151, 717
  - ExternalizeL(), 154–156
  - InternalizeL(), 156–157
  - overloading NewL() and NewLC() to take a read stream, 157
  - RReadStream, 153–154
  - RWriteStream, 152–153
- Stress testing, 668
- String resources, 250–251
- Struct, 250, 253, 302, 679
- STRUCT, 254
- Struct:
  - defined, 717
- Structural testing, 667–668
- Styles, editors, 416–417
- Subsessions, 169–170
- Symbian, defined, 8, 13, 717
- Symbian Installation System (SIS), 27, 50–58
  - Certificate Generator, 50–52
  - Installation File Generator, 50–52
  - makekeys.exe utility, 51–52
  - makesis.exe utility, 51–52
  - .pkg file format, 52–57
  - .sis file:
    - build tools, 51–52
    - building, 58
  - sisar utility, 51–52
- Symbian OS, 5–7, 9
  - architecture, 71
  - as asynchronous operating system, 126
  - cleanup stack, advanced use of, 96–97
  - defined, 717
  - export definitions, 72
  - filenames, 143–144
  - files, 143–151
    - RFile API, 147–151
    - RFs API, 144–147
    - Symbian OS filenames and pathnames, 143–144
  - fundamentals, 75–171
    - client-server architecture, 165–170
    - collection classes, 109–126
    - construction methods, 96
    - descriptors, 97–109
    - pathnames, 143–144
    - stores, 77, 157–164
    - CDirectFileStore, 159–160

**Symbian OS, *continued***

- defined, 157
- reading from, 162–163
- steps in using, 163–164
- Stream Dictionary, 161–162
- types of, 158
- streams, 77, 151–157
  - defined, 151
  - ExternalizeL(), 154–156
  - InternalizeL(), 156–157
  - overloading NewL() and NewLC() to take a read stream, 157
  - RReadStream, 153–154
  - RWriteStream, 152–153
- support for producing devices using non-English languages, 43
- Symbian settings** tab, 58
- SymbianOsUnit, 673
- Syntax highlighting, 63
- Syntax, resource files, 245–247
- System Agent (SA), 651–653, 716
- System capabilities, accessing, 618, 649–654
  - Hardware Abstraction Layer (HAL), 649–651
  - phone IMEI number, 651
  - System Agent, 651–653
  - vibration API support, 653–654
- System testing, 666–667
- System watchdogs, 174, 219, 718
  - and good application behavior, 219
- SystemAgent** example application, 619
- SYSTEMINCLUDE, 32
- System-initiated redraws, 243

**T**

- TAny, 83
- TARGET, 32
- Target device:
  - building, 4
  - deploying, 4
- Target Series 60 device:
  - building for, 19–21
    - C++Builder X, 20
    - CodeWarrior IDE, 20
  - deploying on, 21–22
  - running the application on, 22
  - .sis file:
    - building, 21–22
    - installing, 22
- TARGETPATH, 32
- TBool, 83
- TBrushStyle, 567, 574
- TBuf, 99, 105–106
- TBufBase, 99
- TBufC, 99, 105–106
- TBufCBase, 99
- TChar, 83
- T-classes**, 78–79
- TCP (Transmission Control Protocol), 465, 471, 492, 718
- TCP/IP (Transmission Control Protocol/Internet Protocol), 430, 453, 457–465, 487, 491, 494
  - CommDB, 458–459
  - IPv6, 458
  - multihoming, 459–467
  - programming for Series 60, 458
  - programming infrared on a Series 60 device, 467–470
    - IAS queries, 468–469
    - IrDA Serial API, 469
    - IrDA Sockets API, 467–468
    - IrOBEX (Infrared Object EXchange), 470
    - IrTranP, 469–470
- TcpiplmpEx, 431
- TcpiplmpEx, 431
- TDes, 99, 106–108
- TDesC, 99, 106–108
- TDesC&, 106
- TDesC8, 108–109
- Telephony, 490, 543–551
  - ETel API, 544
  - getting started, 544–545
  - last calling number, retrieving, 550–551
  - making a call, 546–548
    - checking if caller has hung up, 547
    - handling end of call, 548
    - opening RCall and dialing, 547
    - watching for end of call, 547
  - receiving a call, 548–551
    - attempting to answer call, 549
    - handle end of call, 549–550
    - waiting for an incoming call, 549
    - waiting for call to end, 549
- Telnet, 457, 718
- Test harnesses, 681–684
  - RTest, 682–684
- Testing, 658, 664–686
  - black-box, 667
  - debug output, 674
  - file logging, 675–677
  - functional, 667–668
  - integration, 666–667
  - out-of-memory, 684–686
  - performance, 668
  - recovery, 669
  - scripted, 672–673
  - serial output, 674–675
  - strategies for, 666–669
  - stress, 668
  - structural, 667–668
  - system, 666–667
  - on target vs. emulator, 677–681
    - directory differences, 680–681
    - hardware limits, 678–679
    - heap and stack sizes, 678

- machine word alignment, 678–679
  - out-of-disk errors, 679
  - thread process model, 678
  - timing differences, 680
  - writable static data, 680
  - test executables, running, 686
  - test harnesses, 681–684
    - RTest, 682–684
  - test teams/procedure, 666
  - tools/techniques for, 669–677
    - console applications, 672, 707
    - dynamic tools, 670
    - resource failure methods, 672–673
    - static tools, 670–671
    - unit, 666–667
    - white-box, 667, 720
  - Testing** example application, 659
  - TestQuest Pro, 673
  - Testwell CTC++, 672
  - TEventCode, 263
  - TEXT, 254
  - Text, *See* Fonts and text
  - Text editor setting item reference, 387
  - Text editors, 396, 397–417
    - avkon\_flags properties, 407–408
    - configuring, 404–409
    - defining a resource, 405–407
    - dimensions/input capacity, 399–400
    - features of, 398–399
    - global editor, 397
    - input case, 401–402
    - input modes, 401, 408
    - instantiating from a resource, 409
    - keypad input, filtering, 400–402
    - mappings to additional characters, providing, 402–403
    - numeric key maps, 402–403, 408
    - plain editor, 397
    - properties, 404
    - resource/classes, 398
    - rich text editor, 398
      - configuring, 410–416
    - special character tables, 403
    - types of, 397–398
  - TFontSpec, 571
  - Themes, *See* Skins
  - Thread, 34, 64, 67, 127–129, 131, 135, 139–140, 718
  - Thumb, 6, 20, 718
  - TIFF (Tagged Image File Format), 559, 718
  - Time and date editor, 425
  - Time editor, 425
  - Time offset editor, 425
  - Time or date editor setting item reference, 389
  - Timers, 220
  - TInt, 83
  - TInt8, 83
  - TInt12, 83
  - TInt16, 83
  - TInt64, 83
  - TinyTP (Tiny Transport Protocol), 470, 718
  - Title pane, 265, 268–272
    - basics, 268–272
    - changing the text in, 269–270
    - using resources, 270–271
    - displaying an image in, 270–271
    - using resources, 272
  - TitlePane** example application, 225, 268–271, 273–274, 276, 284–285
  - TKeyEvent, 263
  - TLitC, 102
  - TLS (Thread Local Storage), 680, 718
  - TLS (Transport Layer Security), 451–455, 487, 718
  - TMsvEntry, 511
  - TMsvId, 511
  - Tones, 609–610
  - TPckg, 109
  - TPckgBuf, 109
  - TPckgC, 109
  - TPenStyle, 567, 574
  - TPoint, 562
  - TPtr, 99
  - TPtrC, 99, 108
  - Traditional Symbian OS control-based
    - architecture, 184–188
    - when to use, 198
  - Transfer-Encoding, 497
  - Transport Secure Layer (TSL), 502
  - TRAP, 84, 86, 89, 93, 718
  - Trap harness, 84, 718
  - Traps, 84–88
  - TReal, 83
  - TReal32, 83
  - TReal64, 83
  - TRealX, 83
  - TRect, 562
  - TRequestStatus iStatus member, 131
  - TSize, 562
  - TSY, 32–33, 545, 718
  - TText, 83
  - TText8, 83
  - TText16, 83
  - TTypefaceSupport, 571
  - TUInt, 83
  - TUInt8, 83
  - TUInt16, 83
  - TUInt32, 83
  - Twips, 564, 718
- ## U
- 
- UART (Universal Asynchronous Receiver-Transmitter), 675, 719
  - UI (user interface), 6, 27, 35, 39, 41, 45, 60–61, 76–77, 101, 131, 133, 139, 141–142, 160, 177–178, 719

UID, 32  
 UID type, 31, 719  
 UID (unique identifier), 31–32, 44, 49, 53, 68, 160–161, 719  
 UID1, 31–32, 719  
 UID2, 31–33, 719  
 UID3, 31–33, 45, 53, 719  
 UIDs (unique identifiers), client MTM API, 516  
 Uikon, 41, 61, 177, 242, 719  
 UIQ, 5  
 UML (Universal Modeling Language), 81, 176, 719  
 Unicode, 13, 20, 76, 83, 98–101, 103–104, 150–153, 247, 254, 438, 448, 674, 688, 718  
 Unit testing, 666–667  
 URI (Uniform Resource Indicator), 491–492, 495, 505, 719  
 URL (Uniform Resource Locator), 397, 403–405, 451, 497, 605, 719  
 User Datagram Protocol (UDP), 443, 457, 502, 719  
 User heap macros, 665  
 User Interface (UI) implementation, 76  
 User::LeaveIfError(), 86–87  
 User::LeaveIfNull(), 87  
 User::LeaveNoMemory(), 87  
 USERINCLUDE, 32  
 UUID (Universally Unique Identifier), 474, 719

## V

vCards, 638, 717  
 Vertical lists, 342–364  
   basic lists, 348–355  
     adding icons to a list, 349–350  
     allowing the list to scroll, 352–354  
     defining a list using resources, 348  
     defining list items using resources, 351–352  
     handling list events, 354–355  
     instantiating a list using resources, 349  
     list item definitions, 352–353  
   dynamic lists, 355–357  
     changing items dynamically in a list, 356–357  
     defining a dynamic list resource, 355  
     setting items dynamically in a list, 355–356  
   finding items in, 347  
   list items/fields, 346–347  
   markable lists, 344, 346, 357–360  
   menu lists, 344, 345  
   multiselection lists, 344, 346  
   pop-up menu lists, 361–364  
   selection lists, 344  
   types of, 344  
   using, 347–362  
 VibraModeStatus(), 654

VibraSettings(), 653–654  
 Vibration API support, 653–654  
 View, 510, 550–551  
   defined, 720  
**ViewManager** example application, 619  
 View-switching applications, 619–624  
   Calendar view switching, 621–622  
   Camera view switching, 622  
   Messaging view switching, 623  
   nonswitchable applications, 623–625  
   Phonebook view switching, 620–621  
   Photo Album view switching, 622–623  
   Profiles view switching, 623  
 VirtualKey keyword, 65–66  
 Volume setting item reference, 386

## W

WAE (Wireless Application Environment), 502–503, 720  
 Wait notes, 311, 316–319  
   defining in resource, 316–317  
   MAknBackgroundProcess-derived class, 318–319  
   wrapper object, constructing and executing, 316–318  
 Waiting dialogs, 291  
 WAP datagrams, sending, 503–504  
 WAP (Wireless Application Protocol), 490, 502–508, 720  
   architecture, 502–503  
   defined, 502  
   Series 60 1.x WAP APIs, 503–506  
     connected WSP session, 504–505  
     connectionless WSP, 505–506  
     WAP datagrams, sending, 503–504  
   Series 60 2.x WAP APIs, 506–508  
     CWAPBoundCLPushService, 507–508  
     CWAPBoundDatagramService, 506–507  
     CWAPFullySpecCLPushService, 507–508  
     CWAPFullySpecDatagramService, 507  
     WAP datagrams, sending, 506  
   Series 60 implementation, 503–508  
   stack, 502  
 Warning note, 311  
   .wav format, 604  
 WBMP (Wireless Bitmap), 559, 720  
 WDP, 502–503, 720  
 Web-safe colors, 60  
 White-box testing, 667, 720  
 Width (metric), 571  
 Win32, 10, 71, 720  
 Window ownership, 228–229

non-Window-owning controls, 228–229  
Window-owning controls, 229  
Window Server, 556–557, 720  
  shortcut keys, 700–701  
Window-owning controls, 229  
Windows, controls, 225  
WINS (Windows Single process), 64, 678, 720  
WINSB, 72, 720  
WINSBW, 14, 72, 720  
WMF (Windows Meta File), 559, 720  
WORD, 254  
Wrapped notes, 310–313  
Write(), 149–151  
WS32 library, 559  
WSP (Wireless Session Protocol), 492, 502–505,  
  720  
WsSession(), 568

WTLS (Wireless Transport Layer Security), 502,  
  720  
WTP (Wireless Transport Protocol), 502–503, 505,  
  720

---

**X**

XML (eXtensible Markup Language), 491, 523, 720

---

**Z**

Zero(), 101

