

# Index

- #define 3–4, 8, 18–19, 27–8, 246
- #endif 27
- #ifdef 27–8
- #include 27–9
- && (AND Boolean operator) 10–11, 14–17
- & 16–17
- &= (and\_eq compound assignment operator) 16–17
- & (address-of/reference operator) 4–7, 15–18, 21, 23–4, 44–7
- \* 4–7, 10–11, 15–17, 23–4, 43–7
- ++ (increment operator) 5–6, 14–17
- += (assignment operator) 16–17, 59–64
- + (addition operator) 14–17
- + (unary plus) 15–17
- ; 249–50
- == (equal comparison operator) 16–17
- = (assignment operator) 16–17, 34–40, 54, 75–6, 118–20
- >= (greater-than-or-equal-to comparison operator) 16–17
- <= (less-than-or-equal-to comparison operator) 16–17
- != (not equal comparison operator) 12, 16–17
- , (comma operator) 16–17
- . (dot operator) 15–17, 35–6
- > (greater-than comparison operator) 16–17
- >> (input operator) 218–20, 226
- << (left-shift bitwise operator) 16–17
- < (less-than comparison operator) 16–17
- ! (NOT Boolean operator) 10–12, 14–17
- << (output operator) 218–20, 226
- >> (right-shift bitwise operator) 14–17
- :: (scope resolution operator) 15–17, 32–6, 44–7, 54–69
- > (arrow operator) 15–17
- (subtraction operator) 15–17
- (unary minus) 15–17
- (decrement operator) 5–6, 14–17
- / (division operator) 16
- a: drive 215
- abld 239–48
- abstract base classes 62–4, 77–8
- abstraction 31–6, 62–4, 77–8
- Accept 236–8
- access rules, classes 41–7, 290–2
- ACS *see* Authenticated Content Signing
- active object framework, components 149
- active objects
  - see also* asynchronous functions
  - asynchronous service requests 151–2, 199–200
  - background tasks 163–5
  - blocks 147–8, 166
  - CActive 150–67
  - cancelled outstanding request 153, 161–3
  - cleanup stack 154–5, 162–3
  - client-server frameworks 199–208
  - common problems 165–7
  - concepts 147–67, 199–208
  - construction 151–2, 165–6
  - CTimer 155–7
  - destruction 154–5, 162–3
  - error handling 153–60

- active objects (*continued*)
  - examples 155–7
  - multiple active objects 149–50
  - non-pre-emptive multitasking model 149–50, 153–7
  - priorities 150, 151–2, 163–5
  - real-time event handling 150
  - request methods 152–3
  - stray signal panics 160, 165–6
  - unresponsive event handling 166
- active scheduler
  - cancelled outstanding request 153, 161–3
  - concepts 149–67
  - creation 158
  - error handling 153–60
  - event-processing wait loop 158–60
  - exceptions 161
  - extension 160–1
  - installation 158–60
  - leave/trap mechanism 153–7
  - starting 158–60
  - stopping 160
  - stray signal panics 160, 165–6
- ADD 18–19, 151–2, 165–6
- AddFileSystem 213
- addition operator (+) 14–17
- address-of operator (&) 4–7, 15–17, 23–4, 44–7
- After 166
- aggregation relationships
  - see also* composition...
  - concepts 49–69
- AllFiles 262–9, 274–80
- Alloc... 99–100, 113–14, 120–7, 187
- ALWAYS\_BUILD\_AS\_ARM 248
- AND (&&) Boolean operator 10–11, 14–17
- and\_eq (&=) compound
  - assignment operator 16–17
- anti-piracy information 272
- APIs 108, 117–20, 208, 287–9
  - compatibility issues 287–9, 290–1
  - descriptors 117–20
  - documented semantics 288–9
  - extensions 290–1
  - stream store APIs 208, 216–27
- Apparc 185–6
- Append 138–9, 144–5
- AppendL 138
- application design, platform security 272–6
- application layer, OSI 231–3
- application servers, concepts 192
- arithmetic, pointers 5–6
- arithmetic operators
  - see also* entries at start of index
  - concepts 14–17
- arithmetic types, concepts 2–3
- arity concepts 14, 61
- ARM processors 240, 246–8
- ARMV5 240, 247–8
- array 21–2
- arrays
  - see also* CArray...; pointers; RArray...; RPointerArray; TFixedArray
  - concepts 1, 5–6, 18, 21–2, 25–7, 133–46, 189
  - decay 21–2
  - definition 5–6
  - dynamic arrays 25–6, 133–46
  - fixed-length arrays 133, 145–6
  - functions 21–2
  - granularities 139–40
  - out-of-range errors 22, 145–6, 189
  - performance issues 137–9
  - searching 140–5
  - sorting 140–5
- ASCII characters 3, 234
- ASSERT 28–9, 138, 143–4, 187–90
  - \_\_ASSERT\_ALWAYS 138, 142–6, 187–90
  - \_\_ASSERT\_DEBUG 187–90
- assertions, concepts 187–90
- Assign 123–4
- assignment operator (+=) 16–17, 59–64
- assignment operator (=) 16–17, 34–40, 54, 75–6, 118–20
- associativity, operators 14–17
- asynchronous functions
  - see also* active objects
  - cancelled outstanding request 153, 161–3
  - concepts 147–67, 179–80, 191–4, 195–6, 199–200, 206–8, 214, 230–1
  - definition 147
  - examples 147
  - TRequestStatus 151–66
- asynchronous service requests 151–2
- At 146, 225
- Att 212
- Attach 183
- Authenticated Content Signing (ACS) 279–80
- auto 25–6
- automatic storage
  - see also* stack
  - concepts 25, 77
- background tasks, active objects 163–5
- backward/forward compatibility 283–4
- base classes
  - see also* inheritance; TDes...
  - concepts 51–69, 77–8, 102, 109–32, 134–6, 178
- BaseConstructL 105–6
- batteries 149
- Begin 146
- BenQ 169
- bin directory 267–9, 275–7
- binary compatibility, concepts 284, 288–9, 294
- binary data, descriptors 108–9, 126–7
- binary operators, concepts 14, 17, 60–1
- Bind 235–8
- bitwise operators 16–17
- bld.inf 239–41, 248
- bldmake tool 239–48

- block braces (curly brackets) 9–10
- block scope, concepts 9–10, 19, 24, 36
- blocking/non-blocking modes, sockets 230–1
- blocks, active objects 147–8, 166
- Bluetooth 229, 232–3, 234–5
- bool 2, 72–3
- Booleans, concepts 2, 72–3
- bootstrap 252
- break 11–13, 143–4
- breaks, compatibility issues 285–6
- BSD 229, 232
- buffer descriptors
  - see also descriptors; TBuf...
  - concepts 109–32
- build tools 239–48
- BUILD\_AS\_ARM 248
  
- C++
  - concepts 1–29, 31–6, 64–9, 107
  - tools 27–9
- C 1, 107, 134, 161
- c: drive 215, 253–4, 267–9
- C (heap-allocated) classes
  - see also heap
  - cleanup stack 92–100
  - concepts 71, 74–8, 92–100, 138–9, 203–8
  - safe construction/destruction 75–6
- C suffixes, naming conventions 75–6, 80, 92–3, 96–9
- CActive 150–67, 199
  - see also active objects
- CActive::Cancel 154–7, 161–3
- CActive::IsActive 161–3
- CActiveScheduler 151–66
  - see also active scheduler
- CActiveScheduler::Add 151–2, 165–6
- CActiveScheduler::Stop 158–61
- Cancel 154–7, 161–3
- CancelAll 237
- CAPABILITY 242–5, 263–6
- capability model 242–5, 260–6, 274–6
  - assigning 263–4
  - concepts 260–6, 274–6
  - discrete and orthogonal capabilities 260–6
  - rules 264–6
  - system capabilities 261–6
  - TCB capability 263
  - user capabilities 260–1
- CApaDataRecognizerType 185–6
- Carbide.c++ 247–8, 251
- CArray... 134–46
- CArrayFixBase 134–46
- CArrayFixFlat 134–46
- CArrayFixSeg 134–46
- CArrayPakBase 134–46
- CArrayPakFlat 134–46
- CArrayPtrSeg 134–46
- CArrayVarBase 134–46
- CArrayVarFlat 134–46
- CArrayVarSeg 134–46
- case 11–12, 20, 143–4
- catch 84, 86
- CBase, concepts 75–8, 92–3, 95–6, 105–6, 146, 286
- CBufBase 134–8
- CBufFlat 134–6
- CBufSeg 134–6
- CBufStore 220–6
- CDirectFileStore 220–6
- CEikDialog::ExecuteLD 96
- CEmbeddedStore 220–6
- CFileStore 220–6
- char 2–3, 61, 110, 126–7
- character size 107–8, 118, 128–9
- charconv.lib 129
- classes
  - see also C...; M...; R...; T...;
  - types
  - access rules 41–7, 290–2
  - arrays 133–46
  - client–server frameworks 194–209
- compatibility issues 284–6, 290–4
- concepts 9, 31–47, 49–52, 71–81, 290–4
- data storage 35–6
- declarations 31–6, 42–3
- definitions 31–6
- design issues 49–69
- members 36–47, 286–7, 290, 293–4
- naming conventions 71–81
- nested classes 42–3, 46–7, 66–7
- properties 33–5, 49
- purposes 33, 49–52
- relationships 40–7, 49–52
- reuse issues 49–52
- scope 9
- template classes 65–9
- cleanup stack
  - active objects 154–5, 162–3, 212–27
- C (heap-allocated) classes 92–100
- concepts 78, 91–100, 101–6, 124–5, 154–5, 162–3, 204–6, 212–27
- creation 98–9
- item-removal timing decisions 94–5
- pointer decisions 95–6, 104–5
- T/R/M classes 96–9
- uses 92–4
- CleanupArrayDeletePushL 98–9
- CleanupClosePushL 78, 98–9, 124–5, 204–6, 212
- CleanupDeletePushL 97–9
- CleanupReleasePushL 78, 97–9
- CleanupStack 77–8, 87–8, 92–100, 104–5, 124–5, 204–6, 218–19, 223–6
- CleanupStack::Pop 87–8, 92–9, 104–5, 218–19
- CleanupStack::PopAndDestroy 92–9, 124–5, 204–6, 223–6

- CleanupStack::PushL 77–8, 87–8, 92–9, 204–6, 218–19
- client–server frameworks
  - see also* CServer...;
  - RSessionBase
  - active objects 199–208
  - classes 194–209
  - concepts 76, 181–2, 191–209, 211–27, 229–38, 256
  - custom data transfers 202–8
  - data transfers 201–6
  - fundamentals 192–4
  - Hercules example 201–8
  - impact 206–8
  - implementation 191–209
  - overheads 206–8
  - performance issues 206–8
  - R (resource) classes 76
  - read–write request parameters 205–6
  - request arguments 197–8
  - security issues 191–2, 195–6, 197, 199–200
  - sockets 229–38
  - startup 192, 200–1
  - subsessions 207–8
- client–server pattern, concepts 191–2
- Close 76–8, 92, 98, 123–4, 137, 196–208, 212–27, 233–8
- ‘closed’ phones 256–7
- CnvUtfConverter 129–32
- codes, errors 89–90, 96
- CodeWarrior 247, 251
- comma operator (,) 16–17
- CommDD 262–6, 278–80
- commercial developers 272–4
- CommitL 208
- communication protocols, sockets 230–8
- compatibility issues
  - allowed changes 290–4
  - best practice 292–4
  - binary compatibility 284, 288–9, 294
  - breaks 285–6
  - classes 284–6, 290–4
  - concepts 283–94
  - disallowed changes 285–90
  - forward/backward compatibility 283–4
  - levels 283–4
  - libraries 284–5, 292
  - objects 285–6
  - removed items 286
  - source compatibility 284, 288–9
  - virtual functions 287–8, 292–4
- compilation 3–4, 9, 17–18, 27–9, 38–40, 57–69, 83–4, 246–8
  - concepts 27–9, 38–40, 57–69, 83–4
  - constructors/destructors 39–40
  - errors 38
  - linking 28–9
  - separate compilation 28
  - static polymorphism 65–9
- compile-time polymorphism *see* static polymorphism
- Complete 199
- component description file 239–48
- composition relationships
  - see also* aggregation...
  - concepts 49–69
- concrete classes 160–1
- conditional statements
  - see also* if...; switch...;
  - concepts 9–11, 17
- configuration, platform security 276–80
- Connect 105, 193–208, 233–8
- connection-oriented
  - communication protocol, concepts 230–3, 235–6
- connectionless communication
  - protocol, concepts 230–3, 235–6
- const, concepts 3–4, 7–8, 18, 20–1, 39, 45–7, 67–8, 110–11, 125–6, 175–6, 287–9, 291–2
- const pointers, concepts 7–8
- constants 3–4, 7–8, 11–12, 38–40, 45–7, 67–8, 286–9, 291–2
- ConstructL 95–6, 102–6
- constructors
  - see also* copy...; default...;
  - member functions
  - active objects 151–2, 165–6
  - C classes 75–6
  - concepts 34–40, 52–69, 74–6, 88–9, 101–6
  - leave/trap mechanism 88–9, 102, 105
  - T classes 74, 105
  - two-phase construction 101–6
- containers
  - see also* arrays
  - concepts 13–46
- context switches, concepts 148, 180–1, 207–8
- continue 13
- conversions
  - concepts 2–3, 21–2, 37–8, 61, 128–32
  - descriptors 128–32
- Copy 118–20, 128–9
- copy constructors
  - see also* constructors
  - concepts 38–40, 75–6, 111–12
- Count 146
- coupling 51
- CPermanentFileStore 220–6
- CPersistentStore 220–6
- CPolicyServer 199–201
- cpp file extensions 28, 172
- Create 76–7, 114–15, 123, 178–81, 213–27
- CreateL 123
- CreateLC 223–4
- CreateMax 123
- CreateSession 195–205
- CreateSIS 277–8
- critical sections
  - see also* synchronization
  - code 180
- CSecureStore 220–6
- CServer2 199–201

- CSession2 198–208
- CStoreMap 224–6
- CStreamDictionary 221–2, 224–6
- CStreamStore 220–6
- CString 107
- CTestClass 188–90
- CTimer 155–7
- curly brackets (block braces) 9–10
- custom data transfers, client–server frameworks 202–8
  
- d: drive 215
- D suffixes, naming conventions 89–90, 96
- data abstraction 31–6
- data caging, concepts 266–9
- data storage, classes 35–6
- data transfers, client–server frameworks 201–6
- data-link layer, OSI 231–3
- datagram socket 230–8
- \_DEBUG 28, 188–90
- debuggers 28–9, 99–100, 145–6, 155–7, 187–90, 247, 251, 274–6
  - assertion macros 187–90
  - concepts 28–9, 187–90
  - log files 274
  - on-target debugging 28–9
- decay, array-to-pointer decay 21–2
- declarations
  - classes 31–6, 42–3
  - concepts 8–13, 17–20, 22, 28–9, 31–6
  - functions 17–20
  - multi-dimensional arrays 22
  - scope 9–10
  - Symbian OS 71–81
- decrement operator (–) 5–6, 14–17
- def file extensions 170, 172, 245, 247, 287
- default 11–12, 20, 24
- default constructors 38–40, 53–69, 104–5, 286
  - see also constructors
- Define 182–4
- definition statements
  - classes 31–6
  - concepts 8–13, 17–19, 28–9, 31–6
- delete 14, 15–17, 25–7, 46–7, 58–64, 78, 91–9, 105–6, 194–208, 212
  - see also dynamic memory allocation
  - concepts 25–7, 91–2, 105–6
- DeleteAll 146
- dereferencing, concepts 4, 10–11, 23–4, 105–6, 126–7
- derived classes
  - see also inheritance
  - concepts 52–69, 77–8, 104–5, 110–32, 287–8, 292–3
  - descriptors 110–32
  - two-phase construction 104–5
- Des 111, 113–14, 119–20
- descriptors
  - see also HBufC...; RBuf...; TBuf...; TDes...; TPtr...
  - APIs 117–20
  - binary data 108–9, 126–7
  - character size 107–8, 118, 128–9
  - concepts 107–32
  - conversions 128–32
  - correct use of dynamic descriptors 120–7
  - definition 107, 108
  - derived classes 110–32
  - function parameters 120
  - inefficiencies 125–6
  - inheritance 116–17
  - literal descriptors 110–19, 123, 126–7, 130
  - memory management 108–9
  - number conversions 129–32
  - packaging objects 130–2
  - text data 108–9
- design issues, classes 49–69
- destructors
  - see also member functions
- active objects 154–5, 162–3
- C classes 75–6, 154–5
- concepts 38–40, 46–7, 73–7, 88–9, 96–9, 101–6, 155–6
- leave/trap mechanism 84, 88–9
- R (resource) classes 76–7
- T classes 73–4, 84–5
- two-phase construction 105–6
- developer certificates 278–9
- dialogs, resource files 245–6
- digital signatures, installation packages 249, 277–8
- direct file stores 220–6
  - see also stores
- directives, concepts 27–9
- directories 211–27, 253–4, 267–9
- discrete and orthogonal capabilities, concepts 260–6
- DiskAdmin 262–6, 278–80
- DismountFileSystem 213
- division operator (/) 16
- Dll 174–5
- dll file extensions 170, 193–4, 211, 233
- DLLs see dynamic link libraries
- DNS see Domain Name System
- do, concepts 11–13
- DoCancel 151–66
- documented semantics, APIs 288–9
- domain, sockets 230
- Domain Name System (DNS) 233, 234
- DoRecognizeL 185–6
- DoTaskStep 163–5
- double 2, 72
- doubles, concepts 2, 72
- Drive 212
- drives 215–16
- DRM 262–6, 273, 276, 278–80

- dynamic arrays
  - see also* CArray...; pointers; RArray...; RPointerArray
  - concepts 25–6, 133–46
  - granularities 139–40
  - memory layout 133–5
  - performance issues 137–9
  - searching and sorting 140–5
- dynamic binding 53–4, 57–64
- dynamic descriptors
  - see also* descriptors
  - concepts 113–32
  - correct usage 120–5
- dynamic link libraries (DLLs) 4, 46–7, 169–76, 211, 242–5, 263–6, 285–6
  - capability model 263–6
  - compatibility issues 285–6
  - concepts 169–76, 211, 263–6, 285–6
  - const 4
  - exceptions 173
  - EXEs 170–6, 263–6
  - exporting functions 171–3
  - lookups by ordinal/name 173
  - plug-in DLLs 170–3, 185–6, 213, 233–8
  - polymorphic DLLs 170–6, 177
  - ROM/RAM 177
  - shared-library DLLs 170–6
  - static DLLs 173–6, 177
  - types 170–6, 177
  - UIDs 171
  - writable static data 173–6
- dynamic memory allocation
  - see also* delete; new
  - concepts 25–7, 85–91, 108–9, 113–32, 133–46
- dynamic polymorphism
  - see also* polymorphism
  - concepts 56–64
  - substitution concepts 57–8
- e32base.h 75, 92–3, 151
- e32def.h 20, 27–8, 126–7
- e32std.h 130
- E32USER-CBASE 40 162–3
- E32USER-CBASE 46 160, 165–6
- E32USER-CBASE 47 160
- E32USER-CBASE 71 96–7
- EABI *see* Embedded Application Binary Interface
- ECOM plug-ins 170–1, 185–6, 242–5
- EFalse 72
- efficiency issues 101, 107–17, 125–6, 137–9, 174, 206–8, 216–17
- efile.exe 193–4, 211
- efsrv.dll 193–4, 211
- EKA1 169–70, 174, 176, 178–9, 242
- EKA2 169–70, 174, 178–9, 181, 187, 242
- ELeave 85, 104–5, 158
- else statement 10, 18–19, 34–6, 249–50
- Embedded Application Binary Interface (EABI) 246–7
- embedded stores 225–6
- emulator 1, 68, 169–70, 181, 240, 242–8, 251–4, 274–7
  - see also* epoc...
  - concepts 251–4
  - file system 253–4
  - phone hardware 252–4
  - platform security 274–7
  - processes 181
  - uses 251–2
- encapsulation 31–6, 97–9, 131–2
- encryption 213, 276, 279–80
- End 146, 242–5
- EndTask 163–5
- enum
  - see also* user-defined types
  - concepts 3–4, 8, 18, 19–20, 74, 141–6, 201–2, 246, 286
- EOF 12–13
- EPOC 1, 68, 169–70, 181, 242–3
- EPOC32 1, 68, 169, 240, 246–8, 276–7
- EPOC.EXE 251–4
- EPOC\_HEAPSIZE 245
- epoc.ini 252–4, 276–7
- epocrc 246
- EPOC\_STACKSIZE 244–5
- EPriorityIdle 163–4
- EPriorityStandard 151–2
- equal comparison operator (==) 16–17
- Error 160
- errors 22, 38, 83–100, 130, 153–60, 186–90, 213, 216–17
  - see also* leave/trap mechanism; panics
  - active scheduler 153–60
  - codes 89–90, 96
  - compilation 38
  - file server 216–17
  - out-of-range errors 22, 145–6, 189
  - types 22, 145–6, 189, 216–17
- esock.dll 233
- ESOCK.EXE 233, 258–9
- es\_sock.h 233
- ETEL server 257–9
- Ethernet 231
- ETTrue 72
- evaluation order, operators 14–15
- event handling
  - active objects 147–67
  - concepts 148–50, 152–4
  - definition 148
  - unresponsive event handling 166
- event-driven multitasking, concepts 147–67, 177–8
- event-processing wait loop, active scheduler 158–60
- events
  - concepts 147–67
  - definition 147
- exceptions 1, 8, 13, 17, 36, 46–7, 51, 63, 68–9, 72, 74, 80, 83–100, 139, 150, 173, 232, 268–9, 271–2
  - see also* errors; leave/trap mechanism

- EXEDLL 242–5
- EXEs
  - concepts 176–7, 181, 211, 242–5, 263–6, 271
  - DLLs 170–6, 263–6
- EXEXP 242–5
- ExitCategory 179
- ExitReason 179
- ExitType 179
- explicit 40
- EXPORT\_C 172–3
- exporting functions
  - compatibility issues 287, 291
  - DLLs 171–3
- EXPORTUNFROZEN 245
- expressions, concepts 14–17
- extern 9, 13, 29
- externalization
  - see also* streams
  - concepts 217–27, 287
  
- F32 259
- factory functions, concepts
  - 46–7, 75–6, 79
- file server 192, 193–4, 208, 211–27, 257–9, 263
  - see also* RFile
  - concepts 211–27, 257–9
  - efficiency issues 216–17
  - errors 216–17
  - file names 215–16
  - handle class 213–16
  - session class 211–14
- files
  - data caging 266–9
  - name manipulation 215–16
  - organization concepts 27–9
- FILETEXT 250–1
- Find 140–5
- FindIsq 140–5
- ‘fire and forget’ IPC 184–5
- fixed-length arrays 133, 145–6
- flash memory 211
- flat buffers 133–46
- float 65–7, 72–4
- floating-point types
  - see also* double
  - concepts 2–3, 65–7, 72–4, 252–3
- foo 34–47, 53–69, 101–6, 187
- for loop, concepts 11–13, 144–5
- forward/backward compatibility 283–4
- free storage *see* heap
- friend, concepts 40–7, 54–6, 66–7, 68–9
- fsy file extensions 170
- Function 198
- functions
  - arrays 21–2
  - compatibility issues 286–92
  - concepts 2, 9, 15–25, 33–6, 84–5, 198, 286–92
  - declarations 17–20
  - definition statements 17–19, 28–9
  - inline functions 18–19, 24, 28–9, 292–4
  - ordinal numbers 287
  - parameters 17–25, 67–8, 120
  - pointers 23–4
  - prototype concepts 17–18, 19–20, 24
  - return values 13, 22–3, 59–60
  - scope 9
  - template functions 65–9
  - unspecified parameters 19–20, 24
  - virtual functions 24, 52–69, 77–9, 138–9, 151–66, 173, 287–8, 292–3
- fundamental types
  - see also* floating-point...;
  - integer...; types
  - concepts 1–8, 18–19, 71–3
  
- GCC *see* GNU Compiler Collection
- GCCE 240, 247–8
- generic programming 1
- Get 57, 183
- GetByName 234–8
- GetDir 194–208
- GetProtocolInfo 234–8
- global scope 9
- GNU Compiler Collection (GCC) 247–8
  
- GNU Debugger 28–9
- granularities, dynamic arrays 139–40
- greater-than comparison operator (>) 16–17
- greater-than-or-equal-to comparison operator (>=) 16–17
- GUI applications 98, 169, 245–6
  - see also* S60; Series 80; UIQ
  - resource files 245–6
  
- h file extensions 20, 27–8, 170
- handle classes 179–80, 213–16
- hardware builds, concepts 246–51
- HBuFC... 80, 97–8, 113–32, 203–6
  - see also* descriptors
  - concepts 113–32
  - TDesC 125–6
- HEADER 242–5
- heap
  - see also* C classes; dynamic memory allocation
  - concepts 25–7, 71, 74–7, 80, 85–9, 91–2, 101–6, 137, 178–80
  - descriptors 113–32
  - new (ELeave) 85–6, 95–6, 104–5, 158
  - two-phase construction 101–6
- Hercules example, client–server frameworks 201–8
- hiding 9, 31–6, 41–7
- HTTP 231
  
- iConstVal (42) 37–8
- if statement 9–10, 18–19, 34–6, 46–7, 106, 124–5, 143–5, 157, 159–60, 205, 249–50
  - see also* conditional statements
  - implementation
  - client–server frameworks 191–209

- hiding 9
- inheritance 63–4
- import libraries 92–3, 96–7, 172–3
- IMPORT\_C 92–3, 96–7, 114–15, 121–2, 172–3, 202–3, 289–90, 293
- increment operator (++) 5–6, 14–17
- infrared 229, 232, 234–5
- inheritance
  - see also polymorphism
  - concepts 49–69, 288–9
  - descriptors 116–17
  - interface/implementation issues 63–4
  - multiple inheritance 62–4
  - private inheritance 54–5, 68
  - protected inheritance 55–6
  - public inheritance 52–69
- initialization 2, 3, 13, 18, 34–40, 52–69, 75–6, 88–9, 92–3, 102, 138
  - CBase 75–6
- Initialize 76–7, 88, 92–3, 142–3
- InitializeL 88, 92–3
- inline 19, 28–9
- inline functions, concepts 18–19, 24, 28–9, 292–4
- input 218–20, 226, 291–2
- insecure applications, definition 272
- Insert 139, 140–5
- InsertInOrder 141–5
- InsertIsqL 140–5
- Install 158
- installation
  - active scheduler 158–60
  - concepts 248–51, 256–9, 277–8
  - native software installer 280–1
  - phone hardware applications 248–51
  - platform security 256–9, 276–8, 280–1
  - sis files 248–51, 256–9, 277–8, 280–1
- instantiation
  - see also objects
  - concepts 33–6, 46–7, 75, 102–3, 117–20, 123, 125, 128–9
- int 2, 6, 18, 21–2, 34–6, 72–4
- integers, concepts 2–3, 72–4
- integral types, concepts 2–3, 60–1
- inter-process communication (IPC)
  - see also client–server
  - frameworks; message queues; publish and subscribe
  - concepts 76, 181–5, 191–209, 256
- inter-thread/process data transfers 130–2
- interface/implementation issues, inheritance 63–4
- internalization
  - see also streams
  - concepts 217–27
- Internalize 122
- IP 231–3
- IPC see inter-process communication
- IsActive 161–3
- iteration statements, concepts 11–12
- Java 107, 161
- KAFInet 230
- KDynamicLibraryUid 171
- KERN-EXEC 187
- kernel 148, 169–70, 193–4, 206–8, 251, 256–60, 263
- KErr... 86, 90–1, 153–4, 155–6, 158–60, 171, 179, 181, 184, 188, 196–7, 200, 213–14
- KErrAlreadyExists 213
- KErrArgument 188
- KErrCancel 179
- KErrNoMemory 86
- KErrNone 90–1, 153–4, 155–6, 158–60
- KErrNotFound 158–60, 181, 213–14
- KErrNotSupported 235
- KErrPermissionDenied 184, 200
- KErrServerTerminated 196–7
- KExecutableImageUid 171
- Kill 178–9
- KNullUid 171, 222–3, 243, 270–2
- KProtocolInetTcp 230
- KRequestPending 154, 159–60, 165–6
- KSockDatagram 230
- KSockStream 230
- \_L macro 126–7
- L suffixes, naming conventions 87–8
- LDD 242–5
- leaks, memory 85–9, 99–100, 123–4
- leave 13, 73–100
- leave/trap mechanism 13, 73–7, 80, 83–100, 102–6, 138, 153–7, 189–90, 205–6, 212–27
  - active scheduler 153–7
  - causes 84–6, 189
  - concepts 83–4, 102, 189
  - constructors 88–9, 102, 105
  - destructors 84, 88–9
  - exceptions 83–4
  - member variables 88
  - panics 89, 186, 189
  - working with leaves 86–9
- LeaveScan 87–8
- left-shift bitwise operator (<<) 16–17
- Length 109, 118–20, 146
- less-than comparison operator (<) 16–17
- less-than-or-equal-to comparison operator (<=) 16–17
- lexical analysis 74
  - see also TLex

- lib file extensions 170, 245
- libraries 1, 4, 29, 46–7, 92–3, 96–7, 130, 169–76, 242–5
  - compatibility issues 284–5, 292
  - concepts 169–76, 284–5, 292
  - DLLs 4, 46–7, 169–76
  - Symbian OS Library 118, 130
- LIBRARY 242–5
- lifetime *see* scope
- Lincoln Laboratory MIT 229
- linking, compilation 28–9
- Listen 236–8
- \_LIT macro 110–19, 123, 126–7, 130, 141–6, 187–8, 212, 214
- literal descriptors 110–19, 123, 126–7, 130, 286
  - concepts 126–7, 130
  - memory layout 127
- loc files 246
- localized vendor names 249–51
- LocalServices 261
- Location 261
- Lock 213–15
- log files, debuggers 274
- logical groupings 31–6
- Logon 179
- LogonCancel 179
- long 2–3, 72
- longjmp 84, 86
- loop statements *see* iteration statements
  
- M (interface) classes
  - cleanup stack 96–9
  - concepts 71, 77–9, 96–9
- macros, concepts 18–19, 27, 65, 69, 84, 99–100, 110–19, 123, 172–3, 187–90, 246
- makefile 239–48
- makekeys 277–8
- MakeSIS 248–9, 277–8
- malloc 113–15
- math 18, 79
- MaxLength 110, 118–20
- Mem 79
- member functions
  - see also* constructors; destructors
  - concepts 36–47, 53–69, 88
- member selection
  - operator 15–17, 35–6
  - precedence 15–17
- members
  - classes 36–47, 88, 286–7, 290, 293–4
  - pointers 43–7
- memory
  - see also* heap; RAM; ROM; stack
  - C++ constructors and leaves 102
  - descriptors 107–32
  - dynamic memory allocation 25–7, 85–91, 108–9, 113–32, 133–46
  - efficiency issues 101, 107–17, 125–6, 137–9, 174, 206–8
  - leaks 85–9, 99–100, 123–4
  - overrun problems 22, 107–8
  - two-phase construction 101–6
  - types 25–7
- memory management unit (MMU) 207–8, 257–9
- memory sticks 176, 254
- menu bars, resource files 245–6
- message loop/dispatch, Win32
  - applications 149–50
- message queues 184–5
- messages, client–server
  - frameworks 76, 181–2, 191–209
- MFC 107
- MIME type 185–6
- MIT 229
- mixin *see* M (interface) classes
- MkDir 212
- MkDirAll 212
- MMC 176–7, 254
- MMF 259
- MMP files 171, 239–50, 271
  - see also* project definitions
  - concepts 239–45, 271
  - syntax 241–5
- MMU *see* memory management unit
- Modified 212
- modulus operator 16–17
- MountFileSystem 213
- multi-dimensional arrays, declarations 22
- MultimediaADD 262–6, 278–80
- multiple active objects, concepts 149–50
- multiple inheritance, concepts 62–4
- multitasking model
  - see also* active objects
  - concepts 147–67, 169, 177–8
- mutable 8
- mutexes 180
  - see also* synchronization
  
- Nseries 50
- namespace 31–6
- namespaces 9, 31–6
  - concepts 9, 31–6
  - scope concepts 9
- naming conventions
  - prefixes 71–81
  - suffixes 87–8, 96–9
- native software installer, concepts 280–1
- NCP 231
- nested classes
  - see also* classes
  - concepts 42–3, 46–7, 66–7
- network layer, OSI 231–3
- network sockets 230–2, 258
  - see also* sockets
- NetworkControl 262–6, 279
- NetworkServices 261, 274–6
- neutral classes, concepts 108
- New 14, 15–17, 25–8, 61, 85–6, 95–6, 102–6
  - see also* dynamic memory allocation

- New (*continued*)
  - concepts 25–7, 85–6, 102–3
- new (ELeave) 85–6, 95–6, 104–5, 158
- NewL 75–6, 87–8, 103–4, 121–7, 155–7, 285–6
- NewLC 75–6, 103–4, 121–7, 204–6
- NewMax 121–3
- NewMaxL 121–3
- NewMaxLC 121–3
- non-modifiable buffers 111–13, 122–3
- non-pre-emptive multitasking
  - model, active objects 149–50, 153–7
- NOT (!) Boolean operator 10–12, 14–17
- not equal (!=) comparison operator 12, 16–17
- NotifyChange 212
- NotifyChangeCancel 212
- NTT DoCoMo's FOMA user
  - interface 169–70
- NULL 5, 7–8, 10–11, 28–9, 86, 103–6, 107, 126–7, 129, 216
- null pointers, concepts 5, 7–8, 10–11
  
- object code
  - see also EXEs
  - compilation 27–9
- object-oriented programming (OO) 1, 31–47, 51–2
  - see also encapsulation; hiding; inheritance; polymorphism
  - classes/types 51–2
  - concepts 31–47, 51–2, 292–3
  - support 31–6
- objects
  - see also constructors; destructors
  - active objects 147–67
  - compatibility issues 285–6
  - concepts 31–47, 130–2, 202–8, 285–6
  - definition 33
  - packaging objects in descriptors 130–2
  - properties 33–5, 49
  - two-phase construction 101–6
- observer classes 77–8
- on-target debugging 28–9
- One-Definition Rule (ODR) 13
- Open 76–7, 105, 178–81, 195–208, 213–27, 230, 235–8
- Open Source GNU C++ compiler 247
- open standards 169, 256–7
- Open Systems Interconnection (OSI) 230–2
- OpenLC 224–6
- operator 26, 59–64, 67–8, 85–6, 118–20, 130–2, 139, 146
- operator new 85–6
- operators
  - see also entries at start of index
  - associativity 14–17
  - concepts 4–7, 14–17, 58–64
  - definition 14
  - list 15–17
  - overloading operators 26, 37–47, 58–64, 75–8, 85
  - precedence 14–17, 60–1
  - types 14–17
- OPL 161
- OR Boolean operator 10–11, 16–17
- ordinal numbers, API functions 287
- OSI see Open Systems Interconnection
- out-of-range errors 22, 145–6, 189
- output 218–20, 226, 291
- overloading
  - concepts 26, 37–47, 58–64, 75–8, 85
  - rules 60–1
- overriding, concepts 53–4, 58–64
- overrun problems, memory 22, 107–8
  
- Panic 13, 127, 128–9, 141–6, 178–80, 186–8
- panics
  - concepts 13, 89, 94, 109, 127, 128–9, 141–6, 160–6, 178–80, 186–90, 196–7
  - RThread 186–7
  - stray signal panics 160, 165–6
  - threads 178–9, 186–90, 196–7
- paradigms 1
- parameterized polymorphism see static polymorphism
- parameters
  - compatibility issues 289–90
  - concept 17–25, 67–9, 120, 289–90
  - descriptors 120
  - functions 17–25, 67–8, 120
  - reference parameters 20–1, 289–91
  - sockets 230, 235–6
  - unspecified numbers 19–20, 24
  - value parameters 20–1, 67–9, 289–90
- passwords 276
- PCs, emulator 1, 68, 169–70, 181, 240, 242–8, 251–4
- PDD 242–5
- peer-to-peer communication 229
- PETTRAN 176
- phone hardware
  - application installation 248–50
  - emulator 252–4
- physical layer, OSI 231–3
- PINs 275–6
- pixel sizes 253
- pkg files 248–50, 277–80
- PKI see Public Key Infrastructure
- platform security 183–4, 191–2, 195–6, 197, 199–200, 211, 255–81
  - application design 272–6
  - capability model 242–5, 260–6, 274–6
  - concepts 255–81
  - configuration 276–80
  - data caging 266–9
  - emulator 274–7

- installation 256–9, 276–8, 280–1
- native software installer 280–1
- processes 255–81
- publish and subscribe 183–4
- releases 276–80
- servers 257–81
- SID 243, 267–9
- signed software 258–9, 277–80
- stakeholders 272–4
- threat design 272–6
- trust model 255–9
- UIDs 268–72
- untrusted software 258–9, 277–8
- VID 269–72
- PlatSecDiagnostics 277
- PlatSecDisabledCaps 277
- PlatSecEnforcement 277
- PlatSecEnforceSysBin 277
- plug-in DLLs 170–3, 185–6, 213, 233–8
- PLUGIN 242–5
- pointer descriptors
  - see also* descriptors; TPtr...
  - concepts 107–32
  - memory layout 111
- pointers
  - see also* & (address-of/reference operator); arrays
  - arithmetic 5–6
  - concepts 1, 4–8, 15–18, 23–4, 37–40, 43–7, 62–4, 91–2, 95–6, 104–5, 107–32, 291
  - decay 21–2
  - definition 4
  - functions 23–4
  - members 43–7
  - this 37–40, 44–7, 59–64, 285–6, 291–2
  - zero 5
- polymorphic DLLs, concepts 170–6, 177
- polymorphism
  - see also* inheritance
  - concepts 42, 49–69
  - definition 57
  - dynamic polymorphism 56–64
  - static polymorphism 57, 64–9
- Pop 68, 87–8, 92–9, 104–5, 218–19
- PopAndDestroy 92–9, 124–5, 204–6, 223–6
- POSIX server 192
- postfix operators 14–17
- power-conservation methods 149
- PowerMgmt 262–6
- precedence, operators 14–17, 60–1
- PreferredBufSize 186
- prefix operators 14–17
- prefixes, naming conventions 71–81
- preprocessor 3–4, 27–9, 65–9
- presentation layer, OSI 231–3
- Print 155–6
- priorities, active objects 150, 151–2, 163–5
- private access permissions, concepts 34–6, 37–40, 41–7, 52–69, 142–6, 267–9, 274–6, 289, 290–3
- private inheritance
  - see also* inheritance
  - concepts 54–5, 68
- PrivatePath 268–9
- PRJ\_EXPORTS 240–1
- PRJ\_MMPFILES 240–1
- PRJ\_PLATFORMS 240–1
- PRJ\_TESTMMPFILES 240–1
- processes
  - see also* application...; servers
  - capability model 242–5, 260–6
  - client–server frameworks 192–209
  - concepts 180–5
  - context switches 148, 180–1, 207–8
  - definition 180
  - emulator 181
  - file server 192, 193–4, 208, 211–27
  - inter-process communication (IPC) 76, 181–5, 191–209, 256
  - platform security 255–81
  - programming, best practice 27, 292–4
  - project definitions 239–48
    - see also* MMP files
    - concepts 241–5
  - projectname.mmp 239
  - properties
    - classes 33–5, 49
    - objects 33–5, 49
  - protected access permissions 41–7, 52–69, 290–2
  - protected inheritance
    - see also* inheritance
    - concepts 55–6
  - protocol, sockets 230–8
  - ProtServ 195, 262–6
  - prt file extensions 170
  - Psion 1, 169–70
  - Ptr 109, 116–17, 138, 198
  - public access permissions, concepts 34–40, 41–7, 52–69, 74, 77–8, 102–3, 131–2, 142–6, 291–3
  - public inheritance
    - see also* inheritance
    - concepts 52–69
  - Public Key Infrastructure (PKI) 279–80
  - publish and subscribe
    - see also* RProperty
    - concepts 182–5
    - platform security 183–4
  - pure virtual methods 62–3, 77–9, 151–66, 173
    - see also* M (interface) classes
  - Push 68, 92–9
  - PushL 77–8, 87–8, 92–9, 204–6, 218–19
- R (resource) classes
  - cleanup stack 96–9
  - concepts 71, 76–80, 84–5, 92, 96–9, 105–6, 138–9, 203–8

- destructors 76–7
  - RAM 173, 176–7, 211
  - RArray 76, 134, 136–46
  - RBuf 76, 114–32
    - see also descriptors
    - concepts 114–32
    - construction 122–5
    - usage 122–5
  - RCriticalSection 180–1
  - RDebug 155–7
  - RDesReadStream 218–27
  - RDesWriteStream 218–27
  - RDM see Reliably Delivered Message
  - Read 147, 208, 214–27, 236
  - ReadDeviceData 262–6
  - ReadFileSection 212–27
  - ReadL 124–5, 198, 218–27
  - ReadUserData 261, 274–6
  - real-time event handling, concepts
    - 150
  - ReAllocL 123–5
  - recognizer plug-ins, concepts
    - 185–6
  - Recv 235–8
  - RecvFrom 235–8
  - RecvOneOrMore 235–8
  - reference parameters, concepts
    - 20–1, 289–91
  - references, concepts 1, 4–7, 18, 20–1, 291
  - registration codes 272–4
  - relational database 216
  - Release 78, 92, 248–9
  - release builds, assertion macros
    - 187–90
  - releasing applications, platform
    - security 276–80
  - Reliably Delivered Message (RDM)
    - 232
  - removable media 254, 275–6
  - removed items, compatibility issues
    - 286
  - Rename 212
  - Rendezvous 179–80
  - Replace 213, 222–6
  - ReplaceLC 222–6
  - RequestComplete 154, 158–60, 163–4, 198
  - Reset 54, 76–7, 137, 146
  - ResetAndDestroy 136–46
  - resource 267–9
  - resource files
    - see also rss files
    - concepts 245–6
  - Resume 178, 180–1
  - retailers 273
  - return 13, 22–3, 59–60
  - return values, functions 13, 22–3, 59–60
  - reuse issues, class relationships
    - 49–52
  - RFile 207–8, 211–27
    - see also file server
  - RFileReadStream 217–27
  - RFileWriteStream 217–27
  - RFs 76, 193–206, 211–27, 233, 268–9
  - RHandleBase 178, 196–7
  - RHostRevolver 233–8
  - right-shift bitwise operator (>>)
    - 14–17
  - rls files 246
  - Rmdir 212
  - RMessage2 198
  - RMessagePtr2 179, 198, 203–8
  - RMutex 180–1
  - ROM 173, 176–7, 211, 263–9, 279
  - root stream 220–6
  - RPC 231
  - RPointerArray 134, 136–46
  - RProcess 180–1
  - RProperty 182–4
    - see also publish and subscribe
  - RReadStream 208, 217–27
  - rsc files 246
  - RSemaphore 180
  - RSessionBase 194–208
    - see also client–server frameworks
  - rsg files 246
  - Rsocket 147, 233–8
    - see also sockets
  - RsocketServ 233–8
  - rss files 246
    - see also resource files
  - RStoreReadStream 224–6
  - RStoreWriteStream 223–6
  - RThread 158–60, 178–81, 186–7, 198
    - see also threads
    - concepts 178–81, 186–7, 198
    - panics 186–7
  - RTimer 151–66
  - run-time polymorphism see
    - dynamic polymorphism
  - RunError 153–60
  - RunL 151–67, 199–200
    - see also active scheduler
  - RVCT 247–8
  - RWriteStream 208, 217–27
- S prefixes 74
- S60 50, 169, 280
  - scope
    - concepts 9–10, 15–17, 22–3, 25, 31–6
    - levels 9
    - statements 9–10, 17, 22–3
  - scope resolution operator (::)
    - 15–17, 32–6, 44–7, 54–69
  - SDKs 118, 247
  - searching, arrays 140–5
  - secure applications, definition
    - 272
  - Secure Identifier (SID) 243, 267–9
  - SECUREID 243–5, 270–2
  - security policies 184, 191–2, 195–6, 197, 199–200
  - segmented buffers, concepts
    - 133–46
  - self-reference
    - see also this
    - concepts 44–7
  - ‘self-signed’ software 258–9, 277–8
  - semaphores
    - see also synchronization
    - concepts 179–80
  - Send 195–6
  - SendReceive 195–208
  - SendTo 235–8
  - serial communication server 192

- serial ports 192, 253
- Series 60 (Nokia) *see* S60
- Series 80 (Nokia) 169
- Server 199–208
- server sessions, concepts 76–7
- servers 76–7, 148–67, 181–5, 191–209, 211–27, 229–38, 255–9
  - client–server frameworks 191–209
  - concepts 191–209, 211–27
  - ETEL server 257–9
  - file server 192, 193–4, 208, 211–27
  - platform security 257–81
  - sockets 229–38
  - starting 192, 200–1
  - stopping 192
  - window server 192, 257–9
- ServiceL 199
- session layer, OSI 231–3
- sessions
  - client–server frameworks 191–209
  - file server 211–27
- Set 118–20, 183, 216
- SetActive 152–66
- SetAtt 212
- SetDriveName 212
- setjmp 84, 86
- SetLength 118–20
- SetMax 118–20
- SetModified 212
- SetVolumeLabel 212
- ShareAuto 196–7
- shared-library DLLs, concepts 170–6
- ShareProtected 196–7
- short 2–3, 6, 61
- Shutdown 237
- SIBO 1
- SID *see* Secure Identifier
- signed 2–3
- signed software, platform security 258–9, 277–80
- SignSIS 277–8
- singleton class, concepts 36
- sis files 248–51, 256–9, 277–8, 280–1
  - see also* installation
- Size 118–20
- sizeof 15
- Smalltalk 51
- Smartphones 50, 169
- SMB 231
- SMTP 231
- sockets 147, 229–38
  - see also* Rsocket
  - blocking/non-blocking modes 230–1
  - closing 237
  - communication protocols 230–8
  - concepts 229–38
  - definition 229
  - initialization 235–6
  - network sockets 230–2, 258
  - opening 235–6
  - OSI reference model 230–2
  - parameters 230, 235–6
  - protocol modules 233–5
  - reading 236–8
  - Symbian OS architecture 232–5
  - terminology 229–32
  - typical properties 229–30
  - using 235–8
  - writing 237
- software
  - see also* installation
  - platform security 258–9, 276–81
- Sort 140–5
- sorting, arrays 140–5
- SOURCE 242–5
- source code
  - compatibility issues 284, 288–9
  - compilation 3–4, 9, 17–18, 27–8
- SOURCEPATH 242–5
- stack 20–1, 68, 71, 73–4, 77, 78, 84–9, 91–100, 101–6, 112–32, 137, 154–5, 162–3, 178–80, 204–6, 212–27
  - see also* automatic storage; TBuf...
  - cleanup stack 78, 91–100, 101–6, 124–5, 154–5, 162–3, 204–6, 212–27
  - descriptors 112–32
- stakeholders, application design 272–4
- standard template library 1
- Start 158–61
- START RESOURCE 242–6
- StartL 199–208
- StartProtocol 233–8
- StartTask 163–5
- statements
  - concepts 8–13
  - conditional statements 9–11, 17
  - scope 9–10, 17, 22–3
- static 25–7, 45–7, 92–9, 121, 126–7, 142–6, 175–6
- static binding 54, 57–8, 65–9, 103–4
- static classes, concepts 79, 103–4
- static DLLs 173–6, 177
- static polymorphism 57, 64–9
  - see also* polymorphism
- statically strong typing 57–8
- static\_cast 15, 60–4
- STDLIB 161
- Stop 158–61
- storage types 25–7
- stores
  - see also* streams
  - concepts 217, 220–7
  - creation 221–6
  - definition 220
  - embedded stores 225–6
  - reading 224–6
  - swizzles 225–6
- stray signal panics, active objects 160, 165–6
- stream dictionary 221–2
- streams 208, 216–27, 230–8
  - see also* stores
  - concepts 216–27, 235
  - definition 217
- strings 107–32, 286
  - see also* descriptors

- Stroustrup, Bjarne 1, 8, 13, 17, 24, 27, 29, 36, 40, 47, 52, 56, 64, 69
- struct 35–6, 74, 97, 246
  - see also classes
  - T classes 74
- stubbed virtual functions 292–3
- Subscribe 183
- subsessions
  - client–server frameworks 207–8
  - file server 212–27
- substitution concepts, dynamic polymorphism 57–8
- subtraction operator (-) 15–17
- suffixes, naming conventions 87–8, 96
- SupportedDataTyPeL 186
- SurroundingsDD 262–6
- Suspend 178–9, 181
- SwEvent 263–6
- SWInstall 256–9
- switch, concepts 10–11, 20, 143–4
- swizzles, concepts 225–6
- Symbian OS
  - active objects 147–67, 199–208
  - class creation 79–80
  - client–server frameworks 76, 181–2, 191–209, 211–27, 229–38
  - compatibility issues 283–94
  - concepts xxi–xxiv, 1, 27–9, 50, 64–5, 68–9, 105–6, 169–70
  - declarations 71–81
  - descriptors 107–32
  - DLLs 4, 46–7, 169–76
  - dynamic arrays 133–46
  - EKA1 169–70, 174, 176, 178–9, 242
  - EKA2 169–70, 174, 178–9, 181, 187, 242
  - emulator 1, 68, 169–70, 181, 240, 242–8, 251–4, 274–6
  - EPOC 1, 68, 169–70
  - event-driven multitasking 147–67, 177–8
  - file names 215–16
  - file server 192, 193–4, 208, 211–27
  - fundamental types 71–3
  - inter-process communication (IPC) 76, 181–5, 191–209, 256
  - kernel 148, 169–70, 193–4, 206–8, 251, 256–60, 263
  - Library 118, 130, 200
  - lower-level features 169–90
  - naming conventions 71–81
  - open platform 169–70
  - overview 169–70
  - phone hardware 248–50, 252–4
  - platform security 183–4, 191–2, 195–6, 197, 199–200, 211, 255–81
  - processes 180–5
  - resource files 245–6
  - semaphores 179–80
  - sockets 147, 229–38
  - system structure 169–90
  - thin templates 64–5, 68–9, 130–2
  - threads 148–67, 177–85, 186–90, 192–209
  - tool chain 27–9, 239–54
  - two-phase construction 101–6
  - types 71–81, 105–6
  - UIDs 171–3, 182–3, 215, 222–3, 242–51, 268–9
  - v5 107–8
  - v6.0 153, 196–7
  - v6.1 174
  - v7 170
  - v8.0 72, 116, 124–5, 170, 174, 182
  - v8.1 122–3, 124–5, 170, 174–5, 194, 198
  - v9 83, 86, 116, 124–5, 174–5, 183, 185, 239–40, 246–7, 255–6, 270, 276–81
  - writable static data 173–6, 192
- Symbian Signed 258, 270–6, 279–80
- synchronization
  - see also critical sections; mutexes; semaphores
  - concepts 179–81
- synchronous functions, concepts 147–8, 193–4, 214
- syntax 1, 17, 31
- sys directory 211, 267–9, 275–7
- system capabilities, concepts 261–6
- system structure 169–90
- SYSTEMINCLUDE 242–5
- T classes
  - cleanup stack 96–9
  - concepts 71–4, 80, 84–5, 96–9, 105, 202–6
  - constructors 74, 105
  - destructors 73–4, 84–5
- Tait, Field-Marshal, gratuitous inclusion of 126
- TAny 68, 72–4, 92–9, 197–8
- TARGET 242–5
- TARGETPATH 242–5
- TARGETTYPE 171, 242–5
- TBool 72–4
- TBuf... 111–32, 212, 216
  - see also descriptors
- TBufBase 113
- TBufC 111–32
- TBufCBase 112, 116–17
- TCB see trusted computer base
- TCE see trusted computer environment
- TChar 129–30, 175–6
- TCleanupItem 92–3, 97–9
- TCP/IP 230–8
- TCPIP.PRT 233
- TDes base class 109–32, 197, 214
  - see also base classes
- TDesC base class 109–32, 216
  - see also base classes
- HBufC... 125–6
- TDes::operator = 118–20
- telephony server (ETEL server) 257–9

- TEMP 213, 251–2
- template 65–9
- template classes, concepts 65–9
- template functions, concepts 65–9
- template specialization, concepts 67–9
- templates, concepts 1, 64–9
- Terminate 178–9
- ternary operators, concepts 14, 16, 60–1
- text data, descriptors 108–9
- TEXTCONTINUE 250–1
- TFileMode 213–14
- TFileName 125–6, 215–16
- TFixedArray 145–6
- thin templates, concepts 64–5, 68–9, 130–2
- third-party libraries 29
- this
  - see also* self-reference
  - concepts 37–40, 44–7, 59–64, 285–6, 291–2
- thread-local storage (TLS) 174–6
- threads
  - see also* RThread
  - blocked threads 166
  - client–server frameworks 192–209
  - concepts 148–67, 177–85
  - context switches 148, 180–1, 207–8
  - creation 177–8
  - critical sections 180
  - message queues 184–5
  - multiple threads 177–8
  - panics 178–9, 186–90, 196–7
  - running 178–9
  - semaphores 179–80
  - terminations 177–9
- threats, application design 272–6
- throw 73–4, 86
- THUMB 247–8
- TIdentityRelation 141–5
- timer waits, asynchronous functions 147
- timers 252
- TInt... 72–4, 89–90, 113–14, 130, 139, 145–6, 175–6, 202–8, 217–19, 224–5
- TInt8 130, 202–8
- TInt16 130, 202–8, 218, 224–5
- TInt32 130
- TInt64 72–4, 130
- TIPCArgs 196, 197–208
- TKey 140–5
- TKeyArrayFix 140–5
- TKeyArrayPak 140–5
- TKeyArrayVar 140–5
- TLex 74, 129–32
- TLinearOrder 141–5
- TLitC8 126–7
- TLitC16 126–7
- TLS *see* thread-local storage
- TNameEntry 234–8
- TNameRecord 234–8
- tool chain, basics 27–9, 239–54
- tools, C++ 27–9
- TParse 215–16
- TPckg 130–2, 203, 206
- TPckgBuf 130–2
- TPckgC 130–2, 203, 206
- TPoint 175–6
- TPriority 151–66
- TProcessId 180–1
- TProtocolDesc 234–8
- TPtr... 107–8, 110–32, 138, 175–6, 203–8
  - see also* descriptors
- TPtr8 107–8, 138, 203–8
- TPtr16 108
- TPtrC 110–32, 175–6
- transient servers *see* application servers
- transport layer, OSI 231–3
- TRAP 84–96, 138–9, 189, 205–6
- trap mechanism, concepts 89–91
- TRAPD 89–96, 205–6
- TReal 72–4
- TReal32 72, 252
- TReal64 72, 252
- TRequestStatus 151–66, 179–80, 183, 195–6, 206–8
  - see also* asynchronous functions
- trust, platform security 255–9
- trusted computer base (TCB) 211, 256–9, 263, 279, 281
  - capability model 263
  - concepts 211, 256–9, 263, 281
- trusted computer environment (TCE) 256–9
- TrustedUI 263–6
- try 84, 86
- TSecurityPolicy 184, 195–6, 197
- TSpecialCase 200
- TStreamId 223–6
- TSwizzle... 226
- TThreadId 178
- TUId 175–6, 222–3
- TUInt... 72–4, 119, 129–32
- TUInt8 129–30
- TUInt16 129–30
- TUInt64 72–4
- TVersion 202–8
- two-phase construction
  - C (heap-allocated) classes 101–6
  - concepts 101–6
  - derived classes 104–5
  - destructors 105–6
  - phases 102–3
- typedef, concepts 3, 8, 71–4, 108, 113, 125–6
- types
  - see also* fundamental...;
  - user-defined...
  - concepts 1–8, 51–2, 71–81
  - sockets 230, 235–6
  - Symbian OS 71–81
- UDP 231–3
- \_UHEAP\_MARK 99–100

- `_UHEAP_MARKEND` 99–100
- `UI` 29, 50, 98, 166, 192
- UIDs *see* Unique Identifiers
- UIQ 169, 246, 249–51, 280
- unary operators, concepts 14, 15–17, 60–1
- Unicode character sets 107–8, 129, 218
- union 35–6
  - see also* classes
- Unique Identifiers (UIDs) 171–3, 182–3, 215, 222–3, 242–51, 268–72
- Unix 229, 232
- Unlock 213–15
- unresponsive event handling, active objects 166
- unsigned 2–3, 61
- unspecified numbers, parameters 19–20
- untrusted software, platform security 258–9, 277–8
- USB 232, 253
- `User` 79, 127, 141–6, 156, 186–8
- user capabilities, concepts 260–1
- user-defined types
  - see also* `enum`; `typedef`
  - concepts 3–4, 8, 18, 34–6, 40, 49–52
- `User::After` 166
- `UserEnvironment` 261
- `User::Free` 98
- `USERINCLUDE` 242–5
- `User::Leave` 84–90
- `User::LeaveIfError` 84–9, 156, 211–12
- `User::LeaveIfNull` 86
- `User::LeaveNoMemory` 86
- `User::Panic` 127, 141–6, 186–8
- `User::RequestComplete` 158–60, 163–4
- `User::WaitForAnyRequest` 158–60, 162–3, 166
- `User::WaitForRequest` 166
- UTF-7 129
- UTF-8 129
- `utf.h` 129
- utility classes 79–80
- `va_arg` 20
- `va_end` 20
- `va_list` 20
- value parameters, concepts 20–1, 67–9, 289–90
- variables, concepts 2
- `va_start` 20
- vectors 133–4
- Vendor Identifier (VID) 269–72
- `VENDORID` 243, 271
- VID *see* Vendor Identifier
- virtual, concepts 52–4, 58–64
- virtual functions
  - compatibility issues 287–8, 292–4
  - concepts 24, 52–69, 77–9, 138–9, 151–66, 173, 287–8, 292–3
  - stubbled functions 292–3
- virtual methods
  - concepts 53–4, 56–64, 77–8, 173
  - definition 53–4
- virtual overriding
  - see also* dynamic polymorphism
  - concepts 53–4, 58–64
- virtual table, concepts 61–4
- `void` 21–3, 26, 32–6, 41–7, 55–69, 72–4, 92–9, 114–15, 131–2, 141–6, 156, 164–5, 289–93
- Volume 212
- `vptr` 116–17
- `vtable` 287–8, 293–4
- `WaitForAnyRequest` 158–60, 162–3, 166
- `WaitForRequest` 166
- `w_char_t` 2
- web browsers, event handling 148
- Weston, Ian xxi–xxiv
- while loop, concepts 11–13, 20, 144–5
- WiFi 231
- wildcard characters 216
- Win32 applications 149, 181, 251–4
- window server (WServ) 192, 257–9
- `WINSCW` 240, 245, 253–4
- word alignment 253
- writeable static data
  - concepts 173–6, 192
  - definition 175–6
- `Write` 208, 214–27, 236–8
- `WriteDeviceData` 263–6
- `WriteInt8L` 218
- `WriteInt16L` 219
- `WriteL` 198, 218–27
- `WriteReal64L` 218
- `WriteToStreamFileL` 218–27
- `WriteUserData` 261
- `WServ` 257–9
- z: drive 215, 253, 267–9
- zero 118–20
- zero, pointers 5