

Symbols

- ^ operator 298
- operator 296
- ! operator 39, 297
- != operator 39, 220, 297
- (++x) pre-increment operator 300
- (=) assignment operator 299
- (=) operator 299
- (i) icon 285
- (x++) post-increment operator 300
- @avg operator 216
- @catch blocks 211
- @class 99
- @count operator 216
- @distinctUnionOfObjects operator 216–217
- @encode statement 90
- @finally block 211
- @implementation directive 99, 106
- @implementation section 102, 112–113, 117
- @interface declaration 100
- @interface directive 99, 117
- @interface section 106, 109
- @max operator 216
- @min operator 216
- @optional messages 171
- @private attribute 101–102
- @property attributes 111
- @property declarations 14
- @property directive 109, 112–113
- @property statement 110, 114
- @property syntax 109–112
 - method naming 111
 - setter method semantics 111
 - thread safety 111–112
 - writeability 111
- @selector(...) directive 168
- @sum operator 216
- @synthesize 214
- @synthesize directive 112–113
- @throw directive 208, 210
- @try block 211
- * character 59–60
- * operator 296
- *.h file extension 12
- *.m file extension 12
- *.xib files 15
- / operator 296
- & operator 59, 298
- && operator 39, 297
- #import statements 122
- #include statements 122
- % character 40, 42
- % operator 296
- %@ placeholders 132
- %d placeholders 132
- + button 21
- + operator 69, 296
- += assignment operator 299
- < > operator 220
- < operator 39, 221, 297
- << operator 298
- <= operator 39, 221, 297
- = operator 220
- =< operator 221
- == operator 39, 61, 69, 91, 220, 297
- => operator 221
- > operator 61, 107
- > operator 39, 220, 297
- >= operator 39, 221, 297
- >> operator 298
- | operator 298

|| operator 39, 297
 ~ operator 298

A

abort() function 254
 abstract factory design 134
 action methods, <UITableViewDelegate> 155–157
 ActionScript 3-based project 322
 actionSheet 157
 actionSheetCancel: method 158
 Ad Hoc distribution 292
 Add Devices button 291
 addEntriesFromDictionary: message 85
 addObject: message 81
 addObserver: selector 198–199
 address value 100
 addresses, obtaining for variables 59–60
 address-of (&) operator 59
 addTask method 249, 251
 addTasksObject: method 251
 addWidget 104
 Adobe Flash 322–323
 Advanced RISC Machine code. *See* ARM
 advert key 217
 advert property 217–218
 age variable 128
 aggregating values 216–217
 ahead-of-time compilation. *See* AOT
 alertView:clickedButtonAtIndex: method 227
 ALL operator 223
 allKeys method 87–88
 alloc class method 115–116, 119
 alloc message 66, 184, 190
 alloc method 127, 163, 165, 182, 192–193
 allocation, combining with initialization 118
 Allocations Trace Template 284
 alloc-based object creation process 67
 allocWithZone: message 190–192
 allProperties object 220, 227
 alternatives 315

- Adobe Flash 322–323
- C and C++ 315
- iPhone SDK 315–320
 - HTML5, CSS3, and other standards 316–317
 - integration with 317–319
 - PhoneGap 319–320
- Lua 320–322
- Mono (.NET) 323
- Objective-C++ 315
- Ruby 320–322

 anError pointer 208
 animate method 147, 150
 animationDidStopSelector 148
 animationNotification method 146

animationNotification protocol 146, 148–149
 animationStopped methods 148
 animationWillStartSelector 148
 ANY operator 223
 AOT (ahead-of-time) compilation 323
 APIs (application programming interfaces) 264–265
 App Store application 322
 App Store profile 292
 appendString method 161
 AppKit framework 5
 Apple Core Data Programming Guide 253
 Apple developer, process for becoming 288
 Apple Inc., adoption of Objective-C by 314
 Apple Operating System. *See* iOS
 Apple Provisioning Portal website 292
 Apple, block-based APIs in 264–265
 Apple-based platforms 316
 application data 228–256

- changing data model 251–253
 - Core Data framework
 - history of 229–231
 - objects 231–232
 - resources 232–234
- error handling and validation 253–256
- performance 253
- PocketTasks application 234–251
 - adding and deleting people 243–246
 - data model 235
 - defining relationships 236
 - managing tasks 246–250
 - master TableView class 240–243
 - model objects 249–251
 - Person entities in pure code 237–240
 - Xcode Core Data template 234–235

 application programming interfaces. *See* APIs
 Application section 9
 application: didFinishLaunchingWithOptions 238–239, 242
 applicationDidFinishLaunching method 133, 138, 150
 applicationDidFinishLaunching: withOptions method 133, 138
 applicationDidReceiveMemoryWarning 193–194
 applications 238–239, 242

- demo, subclassing in 138–143
- iOS. *See* iOS (Apple Operating System) applications
- Rental Manager
 - developing 29–32
 - making data driven 91–94
 - running on iOS device 292
 - sample object 69–73
 - states of, inspecting with breakpoints 23–24
 - with bugs, creating 277

 Applications subfolder 8

- application-specific query 220
- application-specific variant 320
- Approve button 290
- arguments 62
- arithmetic operators 296
- ARM (Advanced RISC Machine) code 322
- array factory method 81
- arrays 48–50, 61, 75–82
 - adding items to 80–82
 - constructing 75–76
 - elements of
 - accessing 76–77
 - searching for 77–78
 - initializing 49
 - iterating through 79–80
 - fast enumeration 80
 - NSEnumerator class 79–80
 - vs. simple types 50
- arrayWithCapacity: factory method 81
- arrayWithContentsOfURL: message 75, 83
- arrayWithObject: factory method 75
- ASCII character chart 37
- aSimpleDynamicMethod selector 174
- Assembly.LoadFrom 324
- Assign attribute 111
- assignment operators 299–300
- asterisk, in variable name 208
- asynchronous task performance 265–275
 - GCD fundamentals 266–267
 - image loading 273–275
 - image search 271–272
 - introduction to GCD 266
 - RealEstateViewer application 267–271
- atIndexPath method 81
- atIndexPath method 242, 249
- attributes Core Data 233–234
- Attributes Inspector option 15
- Attributes Inspector pane 15
- Author element 159
- authors, parsing with NSXMLParser
 - delegate 159–162
- autoboxing 88
- Automatic Device Provisioning check box 291
- autorelease messages 177, 180, 185–187, 192–193
- autorelease object 190
- autorelease pools 184–190
 - adding objects to 185
 - creating new 185–187
 - limitations of 187–190
 - releasing objects in 187

B

- beginAnimations: context: method 147
- BEGINSWITH operator 222

- BETWEEN operator 221
- bFlag variable identifiers 294
- binding, dynamic 166
- bitwise operators 298
- block literal 259–260, 263–264
- __block storage type 261–263, 274
- blocks, syntax of 257–265
 - block-based APIs in iOS frameworks 264–265
 - blocks and memory management 262–264
 - closures 260–261
- Book elements 159
- BOOL data type 39–40
- Boolean truths 39–40
- boxing 88–91
 - nil vs. NULL value vs. NSNull class 90–91
 - NSNumber class 89
 - NSNumber class 90
- boxView 147–148
- Brautaset, Stig 268
- break keyword 304
- break statement 309–311
- breakpoints, inspecting application states with 23–24
- buffer overrun 38
- bugs, creating application with 277
- Build option 21
- Build Phases tab 21
- Build Settings tab 292
- building process 21
- bundle identifier 9
- buttonPressed 172

C

- C and C++ 315
- C libraries 5
- C++ libraries 5
- CABasicAnimation objects 14, 21
- callers, protocol method 147–148
- callHeads method 13–14, 16, 18
- callTails method 13–14, 16, 18
- CAMediaTimingFunction object 21
- Camel case 295
- cApples variable identifiers 294
- caret
 - in block literal 259
 - in syntax of blocks 258
- Cascading Style Sheets. *See* CSS
- categories
 - considerations when using 138
 - extending classes without subclassing 136
- C-based APIs 5, 315
- C-based application 300
- C-based libraries 313
- cellForRowAtIndexPath: method 30–32, 53–54

- Certificate Revocation List 289
- certificates 289–290
- Certificates page 290
- Certificates tab 289
- char * data type 38–39
- char data type 37–38
- characterAtIndex: message 67
- characters, extracting from strings 67–68
- chFoo variable identifiers 294
- chGender 294
- CIL (Common Intermediate Language)-based assemblies 323
- cityMapping dictionary 93
- cityMappings dictionary 197
- CityMappings.plist file 91, 93
- class clusters 57, 134
- class methods, vs. instance methods 104–105
- class_addMethod 173–174
- classes 97–143
 - adding new methods to at runtime 173–174
 - categories
 - considerations when using 138
 - extending classes without subclassing 136
 - clusters 134–136
 - multiple public 135–136
 - reasons for using 134–135
 - custom, adding new class to project 98–99
 - declared properties 109–115
 - @property syntax 109–112
 - dot syntax 113–115
 - synthesizing property getter and setter methods 112–113
 - declaring interface of 99–106
 - header file for CTREntalProperty class 105–106
 - ivars 100–101
 - method declarations 101–105
 - definitions of 57
 - in Rental Manager application 120–123
 - instance variables
 - accessing existing 129–131
 - adding new 127–129
 - making conform to protocol 148–150
 - objects 115–120
 - combining allocation and initialization 118
 - creating and initializing 115–116
 - destroying 119–120
 - init method 116–118
 - overriding methods 131–134
 - providing implementation for 106–109
 - accessing instance variables 106–107
 - method file for CTREntalProperty class 108–109
 - methods 106
 - sending messages to self 107–108
 - sending messages to 63
 - subclassing
 - in demo application 138–143
 - overview of 124–127
 - Classes folder 268
 - classnameWithxxxx naming convention 283
 - clickedButtonAtIndex: method 227
 - CLLocation.h header file 51
 - closures, blocks as 260–261
 - clusters 134–136
 - multiple public 135–136
 - reasons for using 134–135
 - cmd parameter 167, 173
 - Cocoa API (Application Programming Interface) 204–206
 - Cocoa exceptions 210
 - Cocoa frameworks 4–5
 - Cocoa Touch support library 3–4
 - code
 - pure, Person entities in 237–239
 - silent flaws in 116
 - Code Editor window 286
 - code reuse 57
 - Code Sense 295
 - Code Signing section 292
 - code-signing certificates 323
 - Coin Toss game
 - compiling 21–22
 - developing with Xcode tool 7–15
 - creating projects with 9–12
 - description of 8
 - launching 8
 - writing source code 12–15
 - hooking up user interface 15–20
 - test run 21–27
 - controlling debugger 25–27
 - inspecting application state with breakpoints 23–24
 - running CoinToss game in iPhone simulator 24–25
 - selecting destination 22–23
 - coinLandedOnHeads variable 14, 27
 - CoinTossViewController class 12, 196
 - reviewing connections made to and from 20
 - visually forming connection between button control and 17–19
 - CoinTossViewController header file 12, 16
 - CoinTossViewController.m file 13
 - CoinTossViewController.xib file 15
 - collating values 216–217
 - collection-based data structures 215
 - collections 74–94
 - arrays 75–82
 - adding items to 80–82
 - constructing 75–76

- collections (*continued*)
 - elements of 76–78
 - iterating through 79–80
 - boxing 88–91
 - nil vs. NULL value vs. NSNull class 90–91
 - NSNumber class 89
 - NSValue class 90
 - dictionaries 82–88
 - accessing entries in 84–85
 - adding key/value pairs 85–86
 - constructing 82–84
 - enumerating all keys and values 86–88
 - filtering 220
 - making Rental Manager application data driven 91–94
 - colon character 62–63
 - commitAnimations method 148
 - commitEditingStyle:forRowAtIndexPath method 246
 - Common Intermediate Language assemblies. *See* CIL
 - communicating, with objects 62–66
 - comparison operators 39, 296–298
 - compile-time errors 296
 - compile-time type 164
 - compiling, Coin Toss game 21–22
 - compliance, with KVC 213–217
 - accessing properties via 214
 - key paths 215
 - values 215–217
 - computing power, adjusting expectations for 5–7
 - hardware specifications 6
 - unreliable internet connections 7
 - conditional operator 302–303
 - conditional statements 300–305
 - conditional operator 302–303
 - if-else statement 301–302
 - switch statement 303–305
 - conditions, predicate
 - complex 221–222
 - expressing 220–221
 - configureCell: atIndexPath method 242, 249
 - Console section 278
 - constraints, for integral numbers in real world 33–34
 - CONTAINS operator 222
 - containsObject: message 78
 - context-sensitive editor pane 11
 - Continue button, Xcode debugger window 25, 27
 - continue statement 309, 311
 - controls, adding to view 15–16
 - copy attribute 111, 120
 - copy message 262
 - copyWithZone message 190
 - Core Data attributes 233–234
 - Core Data entities 229–230, 232–234, 236
 - Core Data framework
 - error domains 204
 - history of 229–231
 - objects 231–232
 - managed 231–232
 - persistent store coordinator 231
 - resources 232–234
 - attributes 233–234
 - entities 232–233
 - relationships 234
 - Core Data relationships 234
 - Core Data stack 231–232, 237
 - Core Data templates, Xcode 234–235
 - Core Data-based projects 231
 - CoreData.framework 231–232
 - Corona SDK product 320–321
 - count message 75–76, 79
 - Counter class 295
 - CountOfPeople variable 295
 - createBlock function 261
 - CreateMessageForPerson method 184–185
 - createSampleData method 238–239
 - cross-platform play 325
 - CSS (Cascading Style Sheets) 315
 - CSS3 316–317
 - C-style array 75
 - CTFixedLease class 141–143
 - CTFixedTermLease 140
 - CTFontCreateWithName() method 176
 - CTLease class 139–140
 - CTPeriodicLease object 140–142
 - CTPeriodicLease subclass 140–141
 - CTPerson objects 215
 - CTRentalProperty class 164–166, 173–174, 217–219
 - header files for 105–106
 - method files for 108–109
 - curly braces, in block literal 259
 - custom subclasses 231
- D**
-
- Data Model Inspector 235–236
 - Data Modeling tool 235
 - data models 235, 251–253
 - data source 145
 - data types 28–54
 - additional 34–35
 - basic 32–40
 - Boolean truths 39–40
 - integral numbers 32–35
 - char 37–38
 - custom 44–52
 - arrays 48–50

- data types (*continued*)
 - descriptive names for 50–52
 - enumerations 44–46
 - structures 46–48
- displaying and converting values 40–44
 - NSLog function and format specifiers 40–42
 - type casts and type conversions 43–44
- id 58–59
- Rental Manager application 29–32
 - completing 52–54
 - developing 29–32
- database table 234
- data-driven applications, Rental Manager 91–94
- DBController classes 230
- Deactivate Breakpoints option 25
- dealloc message 180, 284
- dealloc methods 14, 93, 119–120, 234, 242
- DEBUG preprocessor symbol 280
- Debugger Console window 31
- debuggers, controlling 25–27
- debugging 276–287
 - controlling memory leaks with Instruments application 281–283
 - creating application with bugs 277
 - detecting zombie objects 284–287
 - NSLog function 278–281
- DebugSample application 278, 285
- DebugSample_Prefix.pch file 279
- declarations, method 101–105
- declared properties 109–115
 - @property syntax 109–112
 - method naming 111
 - setter method semantics 111
 - thread safety 111–112
 - writeability 111
 - dot syntax 113–115
 - synthesizing property getter and setter methods 112–113
- Declared Properties feature 109
- decreaseRentalByPercent: withMinimum method 103–104
- default priority queue 272
- delegate parameter 146, 158
- delegation 166
- dequeueReusableCellWithIdentifier method 152–153
- dereferencing operation 60
- description method 132–134, 136
- desktop-sized screens 318
- destinations, selecting 22–23
- details variable 71
- deterministic behavior 180
- Developer folder 8–9
- Developer/Applications folder 8
- developers, Apple 288
- development tools 4–5
- development, preparing iOS devices for 289–292
- Devices section 291
- Devices tab 291
- devices, iOS
 - preparing for development 289, 292
 - running applications on 292
- dictionaries 82–88
 - accessing entries in 84–85
 - adding key/value pairs 85–86
 - constructing 82–84
 - enumerating all keys and values 86–88
- dictionaryWithContentsOfURL: message 83
- dictionaryWithObjects: forKey method 83
- dictionaryWithObjectsAndKeys: message 83
- dictionaryWithValuesForKeys: method 214
- didFinishLaunchingWithOptions: method 238–239, 242
- didPresentActionSheet: method 157
- didReceiveMemoryWarning message 193–196
- didReceiveMemoryWarning method 194–197
- didSelectRowAtIndexPath: method 249, 285–286
- dispatch_async function 267, 272
- dispatch_get_global_queue function 267
- dispatch_queue_create function 267
- distinctUnionOfObjects aggregate function 217
- Distribution tab 292
- do loop 308, 311
- do while statement 280
- Document Type Definition. *See* DTD
- doesNotRecognizeSelector 169
- DoomEd coding 314
- dot syntax 113–115
- double data type 35–36
- do-while loop 307
- Download button 292
- DTD (Document Type Definition) 158
- DTrace 8
- dumpDataToConsole method 239, 242
- dynamic binding 166
- dynamic typing 163–176
 - dynamic binding 166
 - messaging 166–171
 - handling unknown selectors 169–170
 - methods, selectors, and implementations 167–168
 - sending message to nil 170–171
 - runtime type information 171–174
 - adding new methods to class at runtime 173–174
 - determining if message will respond 171
 - practical uses of 174–176
 - sending messages generated at runtime 171–172
 - static typing vs. 164–165

E

-
- e character 35
 - editButtonItem 246
 - Editor Style button 235
 - element variable 160
 - elements, of arrays
 - accessing 76–77
 - searching for 77–78
 - else keyword 301
 - emulator term 22
 - ENDSWITH operator 222
 - entities, Core Data 232–233
 - enum keyword 45–46
 - enumerateObjectsUsingBlock block-based API 264
 - enumerating
 - all keys and values in dictionaries 86–88
 - fast enumeration 80
 - enumerations 44–46
 - enumerators 79
 - error codes 204
 - error domains 204
 - error handling, of application data 253–256
 - error object 204
 - error variable 204
 - errors, handling 203–210
 - in Cocoa API 204–206
 - in RentalManagerAPI project 209–210
 - NSError objects 206–210
 - silent flaws in code 116
 - escape sequences 37–38
 - evaluateWithObject: message 220
 - exceptions 210–211
 - catching 211
 - throwing 210
 - explicit type conversion 43
 - exponential notation 35
 - expressions
 - operators in 296–300
 - arithmetic operators 296
 - assignment operators 299–300
 - bitwise operators 298
 - comparison operators 296–298
 - precedence of 300
 - predicate
 - parameterizing and templating 223–224
 - using key paths in 222
 - Extended Details pane (Cmd-E) 282
-
- F**
-
- Facebook web application 316
 - factory methods 67
 - failure, of objects to initialize 116
 - FALSEPREDICATE operator 221
 - fancyAddress 170
 - fBusy variable identifiers 294
 - FIFO (first in, first out) 266
 - File menu 9, 29
 - files, for CTRentalProperty class
 - header files 105–106
 - method files 108–109
 - filteredProperties 227
 - filtering, with predicates 219–224
 - complex conditions 221–222
 - evaluating predicate 219–220
 - expressing predicate condition 220–221
 - filtering collection 220
 - predicate expressions 222–224
 - filterUsingPredicate: message 220
 - Finder window 8, 70
 - first in, first out. *See* FIFO
 - FirstViewController class 196
 - Flash toolchain 323
 - Flashbang Studios 324
 - float data type 35–36
 - floating-point numbers 35–36
 - foo property 111
 - FooBar identifiers 294
 - for loop 49, 77, 308, 311
 - for statement 309
 - forKeys, message 83, 85–86, 214, 218
 - format specifiers 40–42
 - forObject, message 86
 - forRowAtIndexPath, method 246
 - forUndefinedKey, message 217–218
 - forwardingTargetForSelector 169
 - forwardInvocation 169–170
 - Foundation framework
 - error codes 205
 - error domains 204
 - Foundation Kit framework 4
 - Foundation.framework 232
 - foundCharacters, method 161
 - Frameworks section, Xcode main window 12, 21
-
- G**
-
- garbage collection 119
 - GCC (GNU compiler collection) 8
 - GCD (Grand Central Dispatch) 265–275
 - GCD fundamentals 266–267
 - image loading 273–275
 - image search 271–272
 - introduction to GCD 266
 - RealEstateViewer application 267–271
 - GDB (GNU debugger) 8
 - gender instance variables 130
 - generateMessage method 286
 - getKey method 213

getRentalPrice 103
 getter methods 112–113, 131
 getValue message 90
 global queue, running block on 266
 GNU compiler collection. *See* GCC
 GNU debugger. *See* GDB
 Google, Image Search API 268
 Grand Central Dispatch. *See* GCD
 graphical plist file editor 91–92

H

handleComplaint method 108
 hard-to-detect-and-diagnose errors 321
 hardware, specifications for iOS applications 6
 header (*.h) file 13
 header files, for CTRentalProperty class 105–106
 Heads button 17
 heightForRowAtIndexPath: method 154
 Hide button 25
 HIG (Human Interface Guidelines) 17
 history of Objective-C 312–314
 adoption by Apple Inc. 314
 origins 313
 popularization via NeXT Inc. 313–314
 house object 174
 HTML5 316–317
 Human Interface Guidelines. *See* HIG
 Hungarian notation 294–295

I

IBAction keyword 20
 IBOutlet keyword 20
 iCount 294
 id data type 58–59, 64, 165
 IEEE 754 Standard format 36
 if statements 27, 77, 117, 152, 286, 310
 if-else statements 301–303
 iFoo variable identifiers 294
 ILP32 programming model 34
 Image Search API, Google 268
 images
 asynchronous loading of 273–275
 asynchronous searches for 271–272
 ImageTableViewController class 267–268
 ImageTableViewController.h header file 269
 ImageTableViewController.m file 269
 immutable array 80
 immutable objects 68
 implementations 167–168
 accessing instance variables 106–107
 method file for CTRentalProperty class 108–109
 methods 106
 sending messages to self 107–108

in keyword 80
 Include Unit Tests check box 10
 increaseRentalByPercent: withMaximum
 method 106, 108, 110
 indexes, in arrays 48
 indexOfObject: message 78
 indexPath.row property 54
 inheritance 57–58
 init methods 116–118, 127–128, 247
 init-based object creation process 67
 initialization, combining allocation with 118
 initWithContentsOfFile: method 93
 initWithPerson method 247
 initWithString: message 66
 initWithURL: method 116
 initWithXYZ message 192
 initWithXYZ: method 117
 initWithZone 191
 inManagedObjectContext: method 239
 insertObject: atIndex method 81
 instance methods 104–105, 131
 instance variables. *See* ivars
 Instruments application, controlling memory leaks
 with 281–283
 int data type 28, 32, 35
 int keyword 294
 integral numbers 32–35
 additional data types 34–35
 char data type 37–38
 constraints in real world 33–34
 floating-point numbers 35–36
 strings 38–39
 Interface Builder, Xcode 4 4
 interfaces, of classes 99–106
 header file for CTRentalProperty class
 105–106
 ivars 100–101
 method declarations 101–105
 internet, unreliable connections 7
 intValue message 89
 iOS (Apple Operating System) applications 3–27
 adjusting computing power and resource
 expectations 5–7
 hardware specifications 6
 unreliable internet connections 7
 block-based APIs in 264–265
 Coin Toss game
 compiling 21–22
 developing with Xcode tool 7–15
 hooking up user interface 15–20
 test run 21–27
 development tools 4–5
 SDK 288–292
 installing 288–289
 preparing device for development 289–292

- iOS Developer account 9
- iOS developer program 289
- iOS Developer Program license 323
- iOS project templates 10
- iOS Provisioning Portal 291
- iOS Simulator 22–23
- iOS-based templates 10
- iOS-powered device 322
- iPad, hardware specifications 6
- iPhone
 - capabilities 5
 - hardware specifications 6
 - screen 6
- iPhone application, running CoinToss game
 - in 24–25
- iPhone Developer program 323
- iPhone Packer 322
- iPhone Safari web browser 316
- iPhone SDK 315–320
 - HTML5, CSS3, and other standards 316–317
 - integration with 317–319
 - PhoneGap 319–320
- isDone attribute 235
- isEqual: message 69
- isKey instance variable 214
- isKey method 213
- isKey variable 213
- isKindOfClass 171
- isMultitaskingSupported property 175
- iterating, through arrays 79–80
 - fast enumeration 80
 - NSEnumerator class 79–80
- ivars (instance variables) 98, 100–101
 - accessing 106–107, 129–131
 - adding new 127–129

J

- java.lang.Object 100
- JavaScript source code 317
- JavaScript-accessible wrappers 319
- JIT (just-in-time) compilation 323
- JSON format 268
- JSON framework 268

K

- kCAMediaTimingFunctionEaseInEaseOut
 - object 21
- key Mono technologies 323
- key paths, using in predicate expressions 215, 222
 - _key variable 213–214
- key/value pairs, adding to dictionaries 85–86

- keyEnumerator message 87
- keys
 - enumerating, in dictionaries 86–88
 - in dictionaries 82
 - unknown 217–218
- keys array 83, 85
- Key-Value Coding Programming Guide 215
- Key-Value Coding. *See* KVC
- Kitten case 305
- KVC (Key-Value Coding) 212–227
 - and NSPredicate class
 - filtering and matching with predicates 219–224
 - sample application 224–227
 - compliance with 213–217
 - accessing properties via KVC 214
 - key paths 215
 - values 215–217
 - handling special cases 217–219
 - nil values 218–219
 - unknown keys 217–218

L

- label control 19
- languages, procedural-based 56
- lastObject message 77
- length message 67, 171
- length property 215
- Let Me Specify Key Pair Information check
 - box 290
- Libraries section 21
- Library section 291
- Library window 15
- lightweight migrations 252
- LIKE operator 222
- Link Binary option 21
- LLVM (Low-Level Virtual Machine) 8
- localizedDescription method 205
- localizedFailureReason method 205
- localizedRecoveryOptions method 205
- localizedRecoverySuggestion method 205
- log 295
- LogAlways 280–281
- LogDebug macro 280
- logical operators 39, 298
- long qualifier 34
- looping statements 305–311
 - controlling 309–311
 - break statement 310
 - continue statement 311
 - do-while statement 307–308
 - for statement 308–309
 - while statement 306–307
- Low-Level Virtual Machine. *See* LLVM

low-memory warnings, responding to 193–200
 overriding `didReceiveMemoryWarning`
 method 194–197
 UIApplicationDelegate protocol 193–194
 UIApplicationDidReceiveMemoryWarningNoti-
 fication notification 197–200
 LP64 programming model 34
 Lua 320–322

M

Mac App Store 289
 main queue 272–273
 main thread 271, 273
 mainBundle method 64
 makeBlock method 263
 malloc_destroy_zone method 192
 managed objects
 context 231
 models 232
 managedObjectContext method 235, 239
 managedObjectModel method 235
 master TableView class 240–243
 MATCHES operator 222
 matching, with predicates 219–224
 complex conditions 221–222
 evaluating predicate 219–220
 expressing predicate condition 220–221
 filtering collection 220
 predicate expressions 222–224
 memory 177–200
 autorelease pools 184–190
 adding objects to 185
 creating new 185–187
 limitations of 187–190
 releasing objects in 187
 controlling leaks with Instruments
 application 281–283
 object ownership 178–179, 192–193
 reference counting 179–184
 determining current retain count 182–184
 releasing object 180–182
 responding to low-memory warnings 193–200
 overriding `didReceiveMemoryWarning`
 method 194–197
 UIApplicationDelegate protocol 193–194
 UIApplicationDidReceiveMemoryWarning-
 Notification notification 197–200
 zones 190–192
 memory fragmentation 190
 memory leak 178
 memory management, blocks and 262–264
 memory maps 59
 memory zones 190
 message forwarding 166, 169

messages
 nonexistent 64–65
 sending
 to classes 63
 to nil 65–66
 to objects 62–63
 to self 107–108
 messaging 166–171
 determining if message will respond 171
 handling unknown selectors 169–170
 methods, selectors, and implementations
 167–168
 sending messages generated at runtime
 171–172
 to nil 170–171
 method callers protocol 147–148
 method declarations 101–105
 method files, for CTRentalProperty class 108–109
 method implementations 106
 method naming category 110
 method swizzling 174
 method_exchangeImplementations 174
 methods 167–168
 <UITABLEVIEWDATASOURCE> 153
 <UITableViewDataSource> 151–153
 <UITableViewDelegate> action 155–157
 <UITableViewDelegate> setter 154–155
 adding new to class at runtime 173–174
 class, vs. instance methods 104–105
 getter
 manual approach to 130–131
 synthesizing 112–113
 naming 111
 overriding 131–134
 setter
 manual approach to 130–131
 semantics 111
 synthesizing 112–113
 Microsoft .NET development platform 323
 Microsoft's Windows Mobile 317
 Minimal overhead 180
 <MKMapViewDelegate> protocol 153
 mobile devices, adapting Cocoa frameworks
 for 4–5
 model objects 249–251
 models
 data 235, 251–253
 managed object 232
 model-view-controller. *See* MVC
 Mono (.NET) 323
 MonoDroid 325
 MonoTouch 324
 msg object 182
 multiple public clusters 135–136
 mutable objects 68

mutableCopyWithZone 190
 MVC (model-view-controller) 229
 MyBlockTest class 263
 myObject object 172
 myProtocol project 146
 myProtocolAppDelegate.m file 149
 myView class 146–147, 150
 myView delegate 150
 myView object 150
 myView.h file 146

N

name attribute 235
 name object 198–199
 name variable 128
 namespaces 295–296
 naming, for variables 293–296
 Camel case 295
 Hungarian notation 294–295
 namespaces 295–296
 native ARM code 323
 Navigation-based Application template 10, 29–30
 nBar variable identifiers 294
 needsConfiguration 115
 .NET applications 323, 325
 .NET CLR runtime environment 323
 .NET-based languages 324
 new addTask method 251
 New App ID button 291
 New File dialog 91, 98
 New File menu option 98
 New Project dialog 9, 29
 New Project option 9, 29
 newMessage variable 286
 NeXT Inc., popularization of Objective-C by 313–314
 nextObject method 79
 NeXTStep GUI concepts 314
 NeXTStep operating system 313
 NeXTStep technologies 314
 .nib files 21
 nil
 sending messages to 65–66, 170–171
 vs. NULL value vs. NSNull class 90–91
 nil constant 62
 nil receiver 170
 nil values 218–219
 nonatomic attribute 112
 NONE operator 223
 nonexistent messages, sending 64–65
 non-NULL pointer 297
 notFoundMarker argument 85
 notFoundMarker message 84–85
 notification handler, using block as 265
 Novell's MonoDevelop IDE 323
 NS namespace 295
 NS prefix 295
 NSArray class 75, 80, 82, 165
 NSArray's filteredArrayUsingPredicate message 220
 NSAutoreleasepool 285
 NSAutoreleasePool class 185–189
 NSAutoreleasePool instances 186
 NSCocoaErrorDomain error domain 204
 NSCreateZone function 190–191
 NSData object 204
 NSDefaultMallocZone function 191
 NSDictionary class 82, 84, 87, 91–92, 208
 NSEnumerator class 79–80
 NSError objects 203–210, 253
 RentalManagerAPI project 206–208
 userInfo dictionary 205–206
 NSExcption class 210
 NSFastEnumeration protocol 80
 NSFetchedResultsController class 228, 241–242, 246, 253
 NSFetchRequest class 240, 242, 253
 NSInvalidArgumentException 223
 NSInvocation class 169–170
 NSLog function 40–42, 278–281
 NSLog message 174
 NSLog operation 131
 NSLog situation 281
 NSLog statement 182, 190
 NSLog-style format string 67
 NSMachErrorDomain error domain 204
 NSManagedObject 232, 247, 249–251
 NSManagedObjectContext 235, 243
 NSManagedObjectModel 235
 NSMutableArray class 76, 80–81, 135
 NSMutableDictionary class 82–85, 218
 NSMutableString element 159–161
 NSMutableString object 180–181
 NSNotFound value 78
 NSNotification object 199
 NSNotificationCenter 197–199
 NSNull class, vs. nil vs. NULL value 90–91
 [NSNull null] statement 91
 NSNumber class 89, 135, 230
 NSNumber wrapper 135
 NSObject class 124, 131–132, 139–141
 NSObject implements 148
 NSObject method 141
 NSObject object 58
 NSObject version 133
 NSObjects 232
 NSOSStatusErrorDomain error domain 204
 NSPersistentStoreCoordinator 235
 NSPOSIXErrorDomain error domain 204

NSPredicate class 213, 219–220, 222–223
 filtering and matching with predicates
 219–224
 sample application 224–227
 NSPredicate-based expressions 222–223
 NSPredicate-based filtering 224
 NSRecycleZone 192
 NSSelectorFromString 168, 172
 NSSet 219
 NSSortDescriptors 239
 NSSQLiteStoreType 232
 NSString class 64–66, 136–138
 NSString object 190, 214–215, 283
 NSString stringWithFormat 223
 NSStringFromSelector method 168
 NSUnderlyingError key 205
 NSValue class 90
 <NSXMLParser> protocol 158
 NSXMLParser class 158–162
 NSXMLParser delegate methods 158–160, 162
 NSZombie Detection 285
 NSZombies feature 284–285
 NULL character 38
 NULL constant 62
 NULL object 170
 NULL reference exception 65, 181, 298
 NULL value, vs. nil vs. NSNull class 90–91
 numberOfCharacters 171
 numberOfComplaints instance variable 108
 numberOfItems variable 77
 numberOfRowsInSection, method 30–31, 53–54,
 277
 numberOfSectionsInTableView 242
 numbers array 311
 numberWithInt 89
 numberWithRentalPropertyDetail method 89

O

objc_msgSend 166–167
 objc_object struct 58
 objcType, message 90
 Object Library option 15
 object message 198–199
 object ownership 178–179, 192–193
 objectAtIndex message 77–79, 84
 objectAtIndex method 123
 objectEnumerator message 79–80, 87
 objectForKey message 84–85, 87, 93
 Objective-C classes 232, 234
 Objective-C developers 276
 Objective-C objects 214, 229, 233
 Objective-C programming language 3–4
 Objective-C statement 219
 Objective-C syntax 62, 281

Objective-C++ 315
 Objective-C-based APIs 5
 Objective-C-based iOS application 319
 object-orientated programming model 4
 object-oriented programming. *See* OOP
 objects 55–73, 115–120
 combining allocation and initialization 118
 communicating with 62–66
 creating and initializing 115–116
 definitions of 56
 destroying 119–120
 id data type 58–59
 init method 116–118
 OOP concepts 56–58
 definitions 56–57
 inheritance and polymorphism 57–58
 vs. procedural-based languages 56
 pointers 59–61
 comparing values of 61
 following 60–61
 memory maps 59
 obtaining address of variable 59–60
 sample application 69, 73
 sending messages to 62–63
 strings 66–69
 comparing 69
 constructing 66–67
 extracting characters from 67–68
 modifying 68–69
 objectsForKeys 84–85
 objectsPassingTest block-based API 264
 observer argument 198
 Offline Application Cache 317
 one-to-many relationship 234
 one-to-one relationship 234
 Online Certificate Status Protocol 289
 OOP (object-oriented programming) 56–58
 definitions
 of classes 57
 of objects 56
 inheritance and polymorphism 57–58
 vs. procedural-based languages 56
 opacity property 14
 OpenGL 315
 OpenGL ES Application template 10
 OPENSTEP 4.2 Desktop 314
 operators, in expressions 296–300
 arithmetic operators 296
 assignment operators 299–300
 bitwise operators 298
 comparison operators 296–298
 precedence of 300
 optional methods,
 <UITABLEVIEWDATASource> 153
 order object 234

Organizer window 291
 overriding methods 131–134
 ownership, of objects 178–179, 192–193

P

parallel thread 271
 parameterizing, predicate expressions 223–224
 parentheses in syntax of blocks 258, 260
 parse method 160
 Parser subclass 159
 Parser_ProjectAppDelegate.m 161
 parsing author with NSXMLParser delegate 159–162
 Pause button 25
 people, adding and deleting 243–246
 PeopleViewController class 240, 246
 performance of application data 253
 performSelector 171–172
 periodicLease method 140
 persistent store coordinators 231
 persistentStoreCoordinator method 235, 252
 Person class 125–126, 128, 130, 132, 250–251
 Person entities, in pure code
 creating 237–239
 fetching 239–240
 Person objects 230
 person.firstName 249
 Person.h file 130, 251
 PersonDetailViewController 243, 254
 pet variable 304
 PhoneGap 319–320
 plist (Property List) schema 76
 PocketTasks 2.xcdatamodel file 252
 PocketTasks application 234–251
 adding and deleting people 243–246
 data model 235
 defining relationships 236
 managing tasks 246–250
 master TableView class 240–243
 model objects 249–251
 Person entities in pure code
 creating 237–239
 fetching 239–240
 Xcode Core Data template 234–235
 PocketTasks.xcdatamodel 234–235
 PocketTasksAppDelegate.h file 237
 PocketTasksAppDelegate.m 234, 242
 pointers 59–61
 comparing values of 61
 following 60–61
 memory maps 59
 obtaining address of variable 59–60
 polymorphism 57–58

post-decrement operations 300
 precedence of operators 300
 predicate conditions 213, 219
 predicates, filtering and matching with 219–224
 complex conditions 221–222
 evaluating predicate 219–220
 expressing predicate condition 220–221
 filtering collection 220
 predicate expressions 222–224
 predicateWithFormat 223
 primitive data types 32
 procedural-based languages, OOP vs. 56
 Product menu
 Build option 21
 Deactivate Breakpoints option 25
 Run option 24
 Project Navigator pane 11, 21
 projects
 adding new class to 98–99
 creating with Xcode tool 9–12
 properties
 accessing via KVC 214
 declared 109–115
 @property syntax 109–112
 dot syntax 113–115
 synthesizing property getter and setter methods 112–113
 properties array 53–54, 75
 Property List file 92
 property.address 114
 PropertyType enumeration 52, 105
 propertyType property 222
 propertyType value 100
 protocol method callers 147–148
 protocols 144–162
 definition of 145–146
 implementing 146–150
 making class conform to protocol 148–150
 protocol method callers 147–148
 important 150–162
 <UIActionSheetDelegate> protocol 157–158
 <UITableViewDataSource> protocol 150–153
 <UITableViewDelegate> protocol 153–157
 NSXMLParser class 158–162
 provisioning
 manually 291–292
 using Xcode 290–291
 Provisioning Portal website 292
 public clusters, multiple 135–136
 publishAd:error: method 208–209
 pure code, Person entities in
 creating 237–239
 fetching 239–240

Q

QuakeEd coding 314
 QuartzCore framework 21

R

raise method 210
 rangeOfString method 72
 Raptor Copter 324
 readonly attribute 111
 readonly property 111
 readonly attribute 111
 RealEstateViewer application 267–271
 RealEstateViewerAppDelegate.h header file 268
 RealEstateViewerAppDelegate.m file 268
 Record Reference Counts check boxes 285
 reference counting 179–184

- determining current retain count 182–184
- releasing object 180–182

 relationships

- Core Data 234
- defining 236

 Release configuration 292
 release message 262
 releasing objects 180–182, 187
 removeAllObjects method 86
 removeObjectAtIndex message 82
 removeObjectForKey message 85
 removeObjectForKey message 85
 removeObserver 199
 Rental Manager application

- classes in 120–123
- completing 52–54
- developing 29–32
- making data driven 91–94

 RentalManagerAPI project 206–210
 RentalManagerAPI.h header file 207
 RentalManagerAPI.m file 207
 RentalManagerAppDelegate class 193
 rentalPerWeek 113
 rentalPrice property 102–103, 107, 109, 218–219
 rentalPrice value 100, 219
 rentalPrice variable 114
 RentalProperty data type 52
 RentalProperty structure 97, 100, 120
 rentalPropertyOfType 118
 removeObjectAtIndex 82
 replaceOccurrencesOfString 137
 required methods,

- <UITableViewDataSource> 151–153

 reserved words 294
 resolveInstanceMethod 173–174
 Resource section, New File dialog 92

resources

- adjusting expectations for 5–7
- hardware specifications 6
- unreliable internet connections 7

 of Core Data framework 232–234

- attributes 233–234
- entities 232–233
- relationships 234

 respondsToSelector method 148, 171, 175
 result property 19
 retain attribute 111, 120
 retain count 179, 182–184
 retain message 262, 283, 286
 retainCount message 182–183
 return statement 259, 286
 reverseObjectEnumerator message 80
 RootViewController class 92, 194–195, 277, 284
 RootViewController.h file 52–53, 92
 RootViewController.m file 30, 53, 71, 93
 Ruby 320–322
 Run button 292
 run loop 186
 Run option 24
 runMemoryTest method 263
 runtime type

- information about 171–174
 - adding new methods to class at runtime 173–174
 - determining if message will respond 171
 - practical uses of 174–176
 - sending messages generated at runtime 171–172
- making assumptions about 164–165

S

Safari browser-based environment 319
 safety, of threads 111–112
 saveAndDismiss method 243
 scientific notation 35
 <script/> element 319
 SDK (software development kit) 229, 315
 second tab 196
 sectionNameKeyPath 242
 security, of threads 111–112
 SEL data type 168
 selector argument 198–199
 selector name 198–199
 selectors 167–170
 self parameter 107
 [self setStatus:nil] 196
 self.rentalPrice 218
 semantics, of setter methods 111
 serial dispatch queue, creating your own 267
 setAddress 109, 166, 168

- setFlag 164
 - setNilValueForKey 219
 - setObject 85–86
 - setRentalPrice method 103, 106, 109, 114
 - setSortDescriptors method 239
 - setter methods
 - <UITableViewDelegate> 154–155
 - semantics 111
 - synthesizing 112–113
 - Setter semantics category 110
 - setValue
 - forKey 86, 214, 218
 - forObject 86
 - forUndefinedKey 217–218
 - setValuesForKeysWithDictionary method 214
 - short qualifier 34
 - shortcuts
 - Alt-Cmd-4 15
 - Cmd-4 21
 - Cmd-B 21
 - Cmd-Option-P 27
 - Cmd-R 29, 31, 54
 - Cmd-Y 25
 - Control-Option-Cmd-3 15
 - Shift-Cmd-N 9
 - Shift-Cmd-Y 31
 - signed qualifier 32–33, 37
 - simple types, arrays vs. 50
 - simulateCoinToss method 14, 24, 26
 - simulator term 23
 - simulators, running CoinToss game in 24–25
 - Singleton design pattern 91
 - software development kit. *See* SDK
 - software runtime environment 5
 - someString object 171
 - source code
 - connecting controls to 17–20
 - writing with Xcode tool 12–15
 - special catch-all case 303
 - specialization 57
 - Split View-based Application template 10
 - SQL code 229
 - SQL SELECT statement 240
 - SQLite 229
 - square brackets 62
 - states, inspecting with breakpoints 23–24
 - static typing, vs. dynamic typing 164–165
 - status property 19
 - status variable 196
 - Step Into button 25
 - Step Out button 25
 - Step Over button 25–26
 - store coordinators, persistent 231
 - strcat function 38
 - strcpy function 38
 - stringByAppendingString message 69
 - stringByDestroyingVowels method 137–138
 - stringByReplacingOccurrencesOfString 63, 68
 - strings 38–39, 66–69
 - comparing 69
 - constructing 66–67
 - extracting characters from 67–68
 - modifying 68–69
 - stringWithFormat message 67, 285
 - stringWithFormat method 104, 118
 - stringWithObject 192
 - stringWithString 192
 - strlen function 38
 - struct box data type 50–51
 - struct keyword 47–48, 52
 - structures 46–48
 - Student classes 126, 128
 - subclassing
 - in demo application 138–143
 - overview 124–127
 - substringFromIndex method 72
 - substringToIndex method 72
 - substringWithRange message 67–68
 - Supporting Files group 70
 - switch statement 303–305
 - syntax 61
 - synthesizing getter methods 112–113
 - system requirements, for installing iOS SDK
 - 288–289
 - System.Reflection.Emit 323
- ## T
-
- Tab Bar Application 292
 - Tab Bar Application template 10
 - tableView 123, 156–157, 242
 - cellForRowAtIndexPath 71, 93, 273, 277, 280, 282
 - commitEditingStyle 246
 - didSelectRowAtIndexPath 249, 285–286
 - numberOfRowsInSection 30–31, 53–54, 153
 - tableView 277
 - TableView class 240–243
 - Tails button 18
 - Target-Action design pattern 172
 - Task class 250
 - Task entity 235
 - tasks, managing 246–250
 - TasksViewController 246, 249
 - Teacher classes 126, 129
 - Teacher init method 129
 - Teacher object 126
 - templates, Xcode Core Data 234–235
 - templating, predicate expressions 223–224
 - Tenants property 215

test runs, Coin Toss game 21–27
 controlling debugger 25–27
 inspecting application state with
 breakpoints 23–24
 running in iPhone simulator 24–25
 selecting destination 22–23

third-party iOS applications 317

thisObject 166

threads, safety of 111–112

Titanium Mobile–based application 320

toolbar buttons 25

tools, development. *See* development tools

transform.rotation property 14

TRUEPREDICATE operator 221

type casts 43–44

type conversions 43–44

type definition 50

type information, runtime 171–174
 adding new methods to class at runtime
 173–174
 determining if message will respond 171
 practical uses of 174–176
 sending messages generated at runtime
 171–172

typedef keyword 50–52, 259

U

UI_USER_INTERFACE_IDIOM 175

<UIActionSheetDelegate> protocol 157–158

UIAlertView class 209, 254

UIApplicationDelegate class 193–194

UIApplicationDelegate protocol 149–150,
 193–194

UIApplicationDidReceiveMemoryWarning-
 Notification 193, 197–200

UIButton class 172, 197

UIDevice class 175

UIImageView class 155, 273

UIKit elements 147, 151

UIKit framework 4–5

UIKit.framework 232

UILabel controls 12–14, 16, 27, 196

UIPrintInteractionController class 175–176

UISlider class 172

UISlider control 223

UITableView class 151, 157, 197, 277, 281–282

UITableView control 30–32, 54

UITableViewCell class 151–153, 155–156, 283

UITableViewCellAccessoryCheckmark 156

UITableViewCellAccessoryDetailDisclosureButton
 156

UITableViewCellAccessoryDisclosureIndicator
 155

UITableViewCellAccessoryNone 155

UITableViewCellAccessoryType parameter 156

UITableViewCellAccessoryTypes 155

UITableViewCells 151, 153, 155

UITableViewCellStyleDefault 31, 151

UITableViewCellStyleSubtitle 54, 152

UITableViewCellStyleValue1 152

UITableViewCellStyleValue2 152

UITableViewController class 240, 246, 267

<UITableViewDataSource> protocol 150–153
 <UITableViewDataSource> optional
 methods 153
 <UITableViewDataSource> required
 method 151–153

UITableViewDataSource protocol 145, 150–151

UITableViewDataSource section 154

<UITableViewDelegate> protocol 153, 155–157
 <UITableViewDelegate> action methods
 155–157
 <UITableViewDelegate> setter methods 154–155

UITableViewStyleGrouped 151

UITableViewStylePlain 151

UITextField IBOutlets 243

UIView objects 172

UIView subclass 145–149, 155

UIViewController class 186, 193–197, 243, 246

UIViews class 194

UIWebView control 319

underscore (_) prefix 214

Unity3D engine 324

unknown keys 217–218

unsigned qualifier 32, 34

Use for Development button 291

user interface, for Coin Toss game 15–20

userCalledHeads parameter 14, 26

userInfo dictionary 205–206, 208, 210

Utility Application template 10

V

validation, of application data 253–256

valueForKey method 213–214

valueForKeyPath message 215

valueForUndefinedKey message 217–218

values
 aggregating and collating 216–217
 displaying and converting 40–44
 NSLog function and format specifiers 40–42
 type casts and type conversions 43–44
 enumerating, in dictionaries 86–88
 key/value pairs, adding to dictionaries 85–86
 nil 218–219
 of pointers, comparing 61
 returning multiple 215

values array 83, 85

valueWithBytes argument 90

- vardic method 75
- variables
 - instance
 - accessing 106–107
 - accessing existing 129–131
 - adding new 127–129
 - naming conditions for 293–296
 - Camel case 295
 - Hungarian notation 294–295
 - namespaces 295–296
 - obtaining address of 59–60
- View menu 15
- view, adding controls to 15–16
- View-based Application template 9–10, 12
- viewDidAppear message 197
- viewDidLoad method 93, 173, 196–197, 249, 286
- viewDidUnload method 195–197
- Visual Studio 323
- VowelDestroyer interface 137

W

- W3C Geolocation API 318
- warnings, low-memory 193–200
- weak-linking support 176
- web-based applications 315–316, 319
- while loop 79, 306–308, 310–311
- white rectangle 15
- willPresentActionSheet method 157
- willSelect 156
- Window-based Application template 10
- Windows-based PC 323
- wireframe box 15
- wireless connectivity 7
- withMinimum method 104

- withObject message 82
- withString
 - method 63, 68
 - options 137
- world-class mobile browser 317
- writability 111
- Writeability category 110
- WWDC intermediate certificate 290

X

- Xcode 4 4
- Xcode Core Data template 234–235
- Xcode Data Modeling tool 235
- Xcode debugger window 25, 27
- Xcode Organizer 291
- Xcode Organizer window 278
- Xcode tool, developing Coin Toss game with 7–15
 - creating projects 9–12
 - description of Xcode tool 8
 - launching 8
 - writing source code 12–15
- Xcode toolset
 - and iOS SDK
 - downloading 289
 - installing 289
 - provisioning using 290–291
- Xcode window 235
- .xib files 21

Z

- zombie objects, detecting 284–287
- zones, memory 190–192

