Inhaltsverzeichnis

Vorwort **13**

Danksagung **17**

1 Machen Sie sich mit Objective-C vertraut **19**

Thema 1 Lernen Sie die Ursprünge von Objective-C kennen 20

Thema 2 Importieren Sie möglichst keine Header in Header 23

Thema 3 Bevorzugen Sie Literale gegenüber gleichwertigen Methoden 27

Thema 4 Verwenden Sie typisierte Konstanten statt der Präprozessordirektive `#define` 32

Thema 5 Verwenden Sie Aufzählungen für Zustände, Optionen und Statuscodes 37

2 Objekte, Nachrichten und die Laufzeit **45**

Thema 6 Grundlagen von Propertys 46

Thema 7 Lesen Sie Instanzvariablen beim internen Zugriff vorzugsweise direkt 54

Thema 8 Wann sind Objekte gleich? 57

Thema 9 Verbergen Sie die Einzelheiten der Implementierung mit Klassenclustern 64

Thema 10 Hängen Sie benutzerdefinierte Daten mit verknüpften Objekten an bestehende Klassen an 69

Thema 11 Die Rolle von `objc_msgSend` 72

Thema 12 Nachrichtenweiterleitung 76

Thema 13 Debuggen Sie undurchsichtige Methoden mit Methodenswizzling 84

Thema 14 Was sind Klassenobjekte? 88

3 Interface- und API-Design 95

Thema 15 Vermeiden Sie Namenskonflikte mithilfe von Namenspräfixen 96

Thema 16 Verwenden Sie einen designierten Initialisierer 101

Thema 17 Implementieren Sie die `description`-Methode 107

Thema 18 Bevorzugen Sie unveränderbare Objekte 112

Thema 19 Vergeben Sie klare und einheitliche Namen 118

Thema 20 Stellen Sie den Namen von privaten Methoden ein Präfix voran 126

Thema 21 Das Fehlermodell von Objective-C 129

Thema 22 Das Protokoll `NSCopying` 134

4 Protokolle und Kategorien 141

Thema 23 Verwenden Sie Delegate- und Datenquellenprotokolle für die Kommunikation zwischen Objekten 142

Thema 24 Zerlegen Sie Klassenimplementierungen mit Kategorien in handhabbare Segmente 150

Thema 25 Stellen Sie Kategorienamen von Drittanbieterklassen immer ein Präfix voran 154

Thema 26 Vermeiden Sie Propertys in Kategorien 156

Thema 27 Verbergen Sie die Implementierungsdetails mit der Klassenerweiterungskategorie 160

Thema 28 Stellen Sie anonyme Objekte mit einem Protokoll bereit 167

5 Speicherverwaltung 171

Thema 29 Reference Counting 172

Thema 30 Vereinfachen Sie das Reference Counting mit ARC 179

Thema 31 Führen Sie in `dealloc` nur die Freigabe von Referenzen und das Aufräumen des Beobachterstatus durch 189

Thema 32 Vorsicht bei der Speicherverwaltung für ausnahmesicheren Code 192

Thema 33 Vermeiden Sie zyklische Verweise mithilfe schwacher Referenzen 195

Thema 34 Verringern Sie die maximale Speichernutzung mithilfe von Autorelease-Blöcken 200

Thema 35 Spüren Sie Probleme bei der Speicherverwaltung mithilfe von Zombies auf 204

Thema 36 Vermeiden Sie die Verwendung von `retainCount` 210

6 Blöcke und Grand Central Dispatch 215

Thema 37 Blöcke 216

Thema 38 Erstellen Sie `typedefs` für gebräuchliche Blocktypen 223

Thema 39 Verringern Sie die Codetrennung durch Handlerblöcke 226

Thema 40 Vermeiden Sie zyklische Verweise durch Blöcke, die auf die Besitzerobjekte verweisen 233

Thema 41 Verwenden Sie zur Synchronisierung Queues statt Sperren 237

Thema 42 Verwenden Sie GCD statt `performSelector` & Co. 242

Thema 43 Wann müssen Sie GCD verwenden und wann Operations-Queues? 246

Thema 44 Nutzen Sie die Vorteile der Plattformskalierung mithilfe von Dispatch-Gruppen 250

Thema 45 Verwenden Sie `dispatch_once` für die threadsichere einmalige Codeausführung 255

Thema 46 Vermeiden Sie `dispatch_get_current_queue` 256

7 Die System-Frameworks 263

Thema 47 Machen Sie sich mit den System-Frameworks vertraut 264

Thema 48 Verwenden Sie Blockaufzählungen statt `for`-Schleifen 267

Thema 49 Verwenden Sie Toll-Free Bridging für Collections mit benutzerdefinierter Speicherverwaltung 274

Thema 50 Verwenden Sie für Caches `NSCache` statt `NSDictionary` 278

Thema 51 Halten Sie Implementierungen von `initialize` und `load` schlank 283

Thema 52 Denken Sie daran, dass `NSTimer` eine starke Referenz hält 289

Index 295