

App-Entwicklung mit Flutter für Dummies

DAS INHALTS- VERZEICHNIS

» Hier geht's
direkt
zum Buch

Auf einen Blick

Über die Autorinnen	9
Einleitung	23
Teil I: Einführung in Flutter	29
Kapitel 1 Flutter und das große Feld der App-Entwicklung	31
Kapitel 2 Startklar machen und rein ins Vergnügen	39
Kapitel 3 Ihre allererste App	47
Teil II: Programmieren mit Dart	55
Kapitel 4 Pfeilschnell programmieren mit Dart	57
Kapitel 5 Bedingte Anweisungen und Schleifen im Griff	83
Kapitel 6 Sammeln und Sortieren – Collections in Dart	89
Kapitel 7 Asynchrone Programmierung – wenn es mal wieder länger dauert	93
Kapitel 8 Vererbung und weitere praktische Dart-Features	97
Kapitel 9 Debugging in Dart – Probleme finden und lösen	105
Teil III: Wir bauen eine App	113
Kapitel 10 Alles ist ein Widget	115
Kapitel 11 Widgets über Widgets – wie werden daraus tolle App-Screens?	129
Kapitel 12 Ein bisschen DIY zwischendurch – Custom Widgets	187
Kapitel 13 Wenn das, dann das – oder das?	195
Kapitel 14 Wo gehts hier lang? Routing in Flutter-Apps	207
Kapitel 15 Mach alles blau – Theming für Ihre App	217
Teil IV: REST und Firebase – externe Daten beziehen und managen	237
Kapitel 16 Schnittstellen anbinden	239
Kapitel 17 Firebase und der Cloud Firestore	275
Teil V: State-Management	299
Kapitel 18 Stein auf Stein – App-Architektur in Flutter	301
Kapitel 19 State-Management	309
Kapitel 20 State-Management mit Bloc und Cubit	319
Teil VI: Testen, builden und veröffentlichen	345
Kapitel 21 Testing – wer, wie, was und wieso, weshalb, warum?	347
Kapitel 22 Der Android-Build	377
Kapitel 23 Der iOS-Build	393

Teil VII: Top-Ten-Teil	401
Kapitel 24 Unsere 10 Lieblings-Widgets	403
Kapitel 25 Unsere 10 Flutter-Tipps und -Tricks	407
Abbildungsverzeichnis	411
Stichwortverzeichnis	417

Inhaltsverzeichnis

Über die Autorinnen	9
Danksagung	9
Einleitung	23
Über dieses Buch	23
Konventionen in diesem Buch	23
GitHub-Repository	23
Flutter-Version	24
Jedes Kapitel allein gegen die Welt	24
Gender-Love	24
Törichte Annahmen über unsere Leserschaft	25
Wie dieses Buch aufgebaut ist	25
Teil I: Einführung in Flutter	25
Teil II: Programmieren mit Dart	25
Teil III: Wir bauen eine App	26
Teil IV: REST und Firebase – externe Daten beziehen und managen	26
Teil V: State-Management	26
Teil VI: Testing, builden und veröffentlichen	26
Teil VII: Top-Ten-Teil	26
Symbole, die in diesem Buch verwendet werden	26
Wie es weitergeht	27
TEIL I	
EINFÜHRUNG IN FLUTTER	29
Kapitel 1	
Flutter und das große Feld der App-Entwicklung	31
Flutter in a Nutshell	31
Alternativen zur App-Entwicklung mit Flutter	33
Native Entwicklung	33
Cross-Platform-Entwicklung	33
PWA-Entwicklung	35
Vorteile von Flutter	35
Wie funktionieren Flutter und Dart?	37
Flutter-API	37
Dart-API	37
Warum Dart?	37
Kapitel 2	
Startklar machen und rein ins Vergnügen	39
Flutter installieren	39
Das Flutter-SDK	40
Android-Setup	41
iOS-Setup (nur Mac)	43
Troubleshooting	44

14 Inhaltsverzeichnis

Entwicklungsumgebung einrichten	45
Für welche Entwicklungsumgebung soll ich mich entscheiden?	45
Einrichtung VSCode	45
Hilfestellung durch VSCode-Erweiterungen	46
Automatische Code-Fixes und Formatierung	46
Kapitel 3	
Ihre allererste App	47
Eine neue Flutter-App	47
Generieren mit dem Wizard	47
Die Ordnerstruktur	48
Android-App starten	50
iOS-App starten	50
Hot-Reload und Hot-Restart	50
Pummel The Fish	52
Mini-Lastenheft	52
Screens	52
Daten	53
Recap: Einführung in Flutter	54
TEIL II	
PROGRAMMIEREN MIT DART	55
Kapitel 4	
Pfeilschnell programmieren mit Dart	57
Die erste Klasse	58
Das ist Typsache	60
Statisches Type Checking – der Analyzer denkt mit	60
Type Inference – die Schreibearbeit können Sie sich sparen!	61
Objekte bauen mit var, final, late oder const	61
var	62
final	62
late	62
late final	62
const	63
const und final – der Unterschied für Leute, die es ganz genau wissen wollen	63
Wie Funktionen funktionieren	64
Aufbau einer Funktion	64
Funktion versus Methode	64
Private Methoden	65
Lambda – die anonyme Funktion	66
Methoden und Funktionen in einer Klasse	66
Externe Funktionen über Packages einbinden	69
Funktionen aufrufen	70
Parameter für jede Lebenslage	70
Positional Parameters	71
Optional Named Parameter	71
Required Named Parameter	72

Alle Parametertypen in Kombination	74
Achtung, statisch!	75
Null oder nicht null – das ist hier die Frage	76
Von null auf hundert	76
In einer Zeit vor Null Safety	77
Null unter Kontrolle	78
Sound Null Safety und Unsound Null Safety – wo ist da der Unterschied?	81

Kapitel 5

Bedingte Anweisungen und Schleifen im Griff **83**

Wenn A, dann B – bedingte Anweisungen in Dart	83
if und else – wenn ich könnte, würde ich ja	83
Switch – wer die Wahl hat, hat die Qual	84
Round and round it goes ... Schleifen in Dart	86
for-Schleife	86
for-in-Schleife – oder auch for-each-Schleife	87
while-Schleife	87
do-while-Schleife	87

Kapitel 6

Sammeln und Sortieren – Collections in Dart **89**

Drei Arten von Collections	89
List	89
Map	90
Set	90
Methoden für Iterables	91

Kapitel 7

Asynchrone Programmierung – wenn es mal wieder länger dauert **93**

Futures, async und await	93
Ein Datenfluss – auch Stream genannt	95
Ein Stream.periodic in der Praxis	95
Ein Stream als Rückgabewert	95

Kapitel 8

Vererbung und weitere praktische Dart-Features **97**

Vererbung in Dart	97
Interfaces	99
Abstrakte Klassen und Methoden	100
Interfaces in der Praxis	100
Mixins	101

Kapitel 9

Debugging in Dart – Probleme finden und lösen **105**

De-BUG-ging – die Jagd auf die Bugs	105
Debuggen in VSCode	106
Mit Breakpoints die Zeit anhalten	107
Die Debug Console	109

Die DevTools von Flutter	109
Widget Inspector – der App-Aufbau	110
Recap: Programmieren mit Dart	111

**TEIL III
WIR BAUEN EINE APP 113**

**Kapitel 10
Alles ist ein Widget..... 115**

Hier fängt alles an: die main.dart-Datei	115
Widgets, Widgets überall	117
StatefulWidget und StatelessWidget.....	119
Auch App-Screens sind Widgets.....	120
StatefulWidget mit setState updaten	122
Wann sollten Sie ein StatelessWidget und wann ein StatefulWidget nehmen?	123
Es wächst ein Widget-Baum.	124
Exkurs: Das Flutter-Framework – vor lauter Bäumen die App nicht mehr sehen	125
Die verschiedenen Ebenen des Flutter-Frameworks	125
Flutter – die Recycling-Maschine	126
Der BuildContext – der Kreis schließt sich	127

**Kapitel 11
Widgets über Widgets – wie werden daraus tolle
App-Screens? 129**

Screens anlegen.	130
Der erste eigene App-Screen	131
Screens importieren	133
Pummel The Fish – der SplashScreen.....	135
Scaffold – ein hübsches Skelett	137
SafeArea – sicher ist sicher.....	137
Container – das Widget für jede Gelegenheit	138
Center – finden Sie Ihre Mitte.....	140
Image – viele bunte Bilder	142
Padding – ein bisschen Polster hat noch nie jemandem geschadet	147
Pummel The Fish – der HomeScreen	148
Die wichtigsten Layout-Widgets.....	148
Die initState-Methode	153
Eine fancy AppBar	154
Columns und Rows – wir bauen einen Screen!	155
FloatingActionButton – und ... Action!.....	156
Icon – this is iconic!	158
ListView.builder – der fleißige Builder nimmt Ihnen die Arbeit ab	158
Pummel The Fish – der DetailPetScreen	160
Einem Screen Daten übergeben	160
Screen Layout mit alten Bekannten	160
Card – das leicht abgehobene Widget.....	163
Aufrufen des Screens mit Datenübergabe.....	164
Stack.....	165

Pummel The Fish – der CreatePetScreen	167
Form – Eingabefelder brauchen auch ein Zuhause	168
TextFormField – hier könnte Ihr Text stehen!	169
DropDownButtonFormField – einmal wählen bitte	173
DropDownMenuItem – Auswahlmöglichkeiten definieren	174
Checkbox – perfekt für Booleans	175
ElevatedButton – Speichern muss sein	178
Eingaben sammeln und Pet-Objekt erstellen	179
Form-Validierung – Vertrauen ist gut, Kontrolle ist besser	183

Kapitel 12

Ein bisschen DIY zwischendurch – Custom Widgets 187

Custom Widget – ja, nein, vielleicht?	187
_CustomWidget – das sollte lieber privat bleiben	188
Ein Widget extrahieren	188
Parameter anpassen	189
Ein Custom Widget für alle und überall!	191
Ein Custom Widget anlegen	191
Der GestureDetector – das Widget mit Fingerspitzengefühl	192
Ich bin je der Ordnung Freund gewesen (Goethe)	193

Kapitel 13

Wenn das, dann das – oder das? 195

Wenn die Daten die UI bedingen sollen	195
Keine Angst vorm Gendern	196
Nach Tierarten unterscheiden	196
Natives Design	199
Responsiveness umsetzen	201
Screen-Größe mit MediaQuery prüfen	202
Ausrichtung mit MediaQuery prüfen	204

Kapitel 14

Wo gehts hier lang? Routing in Flutter-Apps 207

Wie gehts zum nächsten Screen?	207
Navigieren durch eine Timer-Funktion – der SplashScreen	208
Navigieren durch User-Input – der HomeScreen	209
Navigieren mit Back-Button – DetailPetScreen und CreatePetScreen	210
Named Routing – beim Navigieren den Überblick behalten	210
Routen definieren	211
Ich verfolge Sie auf Schritt und Klick – der Backstack	212
Zurück mit Navigator.pop()	213
Bloß nicht zurück – Navigator.pushReplacementNamed()	213
Zurück auch ohne Navigator.pop	214
Routing im Web und für Fortgeschrittene	216

Kapitel 15

Mach alles blau – Theming für Ihre App 217

Wo das Theming in Ihrer App haust	218
Farbe bekennen!	219

Color versus MaterialColor	219
Einzelne Farben – oder gleich die ganze Palette?	220
Farbpaletten vereinfachen die Sache	221
Eigene Farben definieren	222
Das Theme für die »Pummel The Fish«-App einrichten	223
Die »Pummel The Fish«-App farblich nachjustieren	224
Fun mit Fonts	227
Importieren von Fonts.	230
Deklaration der Font in der pubspec.yaml.	230
Definition der Default-Font und einer Handvoll Textstile	231
Textstile in der App verwenden	232
Noch mehr Textstile?	234
Recap: Wir bauen eine App	235

TEIL IV REST UND FIREBASE – EXTERNE DATEN BEZIEHEN UND MANAGEN 237

Kapitel 16 Schnittstellen anbinden 239

Wer oder was ist eigentlich dieser REST? Und was hat er mit API vor?	239
Alternativen zu REST	240
REST-Requests	240
Request-Endpunkt mit 0-n Parametern	241
Request-Methoden	241
Request Headers	242
Request Body	242
REST-Response.	243
Die fünf Klassen der Status-Codes	243
Wenn noch was dranhängt – der Response Body.	244
Los gehts – Daten per REST abrufen.	244
Ohne das http-Package geht hier gar nichts	244
Repositories zur Datenverwaltung	245
Das RestPetRepository	245
Basis-Elemente der REST-API-Anbindung.	246
GET – Daten von einer Schnittstelle abrufen	247
JSON-Daten konvertieren	250
GET-Request triggern – ein Button muss her.	254
POST – Daten an die API senden	255
POST-Request triggern – noch ein Button	257
PUT/PATCH, DELETE – Daten modifizieren.	258
Asynchrone REST-API-Daten im Flutter-UI anzeigen	259
Daten vom Backend holen – aber wann und wo?	260
Kein Problem ohne Ursache.	262
FutureBuilder to the Rescue!	262
Daten aus dem Flutter-UI sammeln und an die REST-API senden	267
Der CreatePetScreen wird funktional	268
Der erste Testlauf.	269

Wo ist mein neu angelegtes Pet?	272
Neuladen der Liste	272

Kapitel 17

Firestore und der Cloud Firestore 275

Die eierlegende Wollmilchsau	275
Cloud-Firestore-Grundlagen	276
Die Vorteile einer Cloud-Firestore-Datenbank	278
Die Nachteile einer Cloud-Firestore-Datenbank	279
Firestore-Installation und Einrichtung	281
Voraussetzungen	281
Firestore CLI installieren	281
Ein neues Projekt anlegen	282
Firestore zur App hinzufügen	283
Cloud-Firestore-Anbindung	284
Vorbereitungen	284
Daten im Firestore speichern	286
Daten in Cloud Firestore einsehen	288
Security Rules für Testzwecke anpassen	289
Daten aus der Firestore-Datenbank auslesen	291
Schön der Reihe nach – Daten filtern und Queries	293
Reaktive Daten – was ist das?	294
Ein StreamBuilder muss her	294
Cloud-Firestore-Alternativen – ja, aber wann und warum?	297
Das Preismodell oder »Hilfe, warum bin ich plötzlich arm?«	297
No NoSQL?	297
Recap: REST und Firestore – externe Daten beziehen und managen	297

TEIL V

STATE-MANAGEMENT 299

Kapitel 18

Stein auf Stein – App-Architektur in Flutter 301

Was ist eine Architektur überhaupt?	301
Das Chaos im Griff mit Ordnerstrukturen	302
In Schichten denken – Struktur durch Layer	302
Die »Pummel The Fish«-Struktur und der Screen-First-Ansatz	303
Der Layer-First-Ansatz	304
Der Feature-First-Ansatz	305
Welcher Ansatz gewinnt?	305
Pragmatisch, praktisch, gut – unser Architektur-Vorschlag	306
Presentation Layer – Screens und Widgets	306
Application Layer – State-Management	306
Data Layer – Models und Repositories	307
Data Provider Layer – FirebaseFirestore, HttpClient und Co.	307
Architektur ist oft ein Prozess	307

Kapitel 19**State-Management 309**

Was ist State-Management? Wozu braucht man das?	309
Lokaler und globaler State	309
Die Grenzen von setState	310
Kurzer Exkurs – das InheritedWidget	312
Wie baut man ein InheritedWidget?	312
Den petCount aktualisieren	313
Ansiedeln des InheritedWidgets im Widget-Baum	315
Das Builder-Widget	316
Warum nicht eine ganze App mit InheritedWidgets bauen?	317

Kapitel 20**State-Management mit Bloc und Cubit 319**

Meine Straße, mein Zuhause, mein Bloc?	319
Warum ausgerechnet das bloc-Package?	320
Kurze Einführung in Bloc und Cubit	320
Ihren ersten Cubit anlegen.	323
Installation.	323
Der ManagePetsCubit	323
Kommunikation zwischen UI und Cubit.	327
Parallelen zum InheritedWidget?	327
BlocProvider – die Ansiedlung im HomeScreen	327
BlocBuilder – States im UI anzeigen	329
Aufrufen von Cubit-Funktionen	331
BlocListener – die benutzende Person informieren	332
BlocListener + BlocBuilder = BlocConsumer	333
BlocSelector – ein Profi-Widget zur Performance-Optimierung	334
Repositories zentral zur Verfügung stellen mit RepositoryProvider	334
Cubits weiter vereinfachen – mit einem Enum-State ans Ziel.	336
Den State als Enum definieren.	336
Die copyWith-Methode – Ihr neuer Begleiter.	337
Anpassungen im Cubit	338
Last but not least – der ManagePetsStatus im UI	339
Mehrere Wege führen nach Rom, welcher ist der richtige?	340
Bloc und Cubit – so unterschiedlich und doch so gleich	341
Traceability	341
Event Transformations	342
Race Conditions in Cubits	342
Good to Know und weiterführendes Wissen.	343
BlocProvider.value	343
States filtern mit listenWhen und buildWhen	343
Recap: State-Management	344

TEIL VI TESTEN, BILDEN UND VERÖFFENTLICHEN 345

Kapitel 21 Testing – wer, wie, was und wieso, weshalb, warum?..... 347

Warum testen?	347
Warum testen, wenn man auch auf Fehler reagieren kann?	347
Test Driven Development	348
Warum wird oft nicht getestet?	348
Wann sind (automatisierte) Tests sinnvoll?	349
Manuell oder automatisiert?	349
Einfach mal durchklicken – manuelle User-Tests	349
Durchklicken lassen – automatisierte Tests	350
Logik testen	350
Unit-Tests in der Theorie	350
Unit-Tests in der Praxis	354
Blocs und Cubits testen	363
User Interface testen	366
Widget-Tests	366
Golden Tests	369
Einen Flow mithilfe von Integration-Tests testen	370
Installation und erste Schritte	370
Der Testfall	371
Integration-Tests laufen lassen	372
Messbarkeit von Tests: die Test-Coverage	372
Es ist nicht alles Gold, was glänzt.	373
Visualisierung der Test-Coverage mit Coverage Gutters	373
Die Test-Coverage im Überblick mit Flutter Coverage	374
Letzte Tipps	375

Kapitel 22 Der Android-Build 377

Vorbereitung eines Builds (Android und iOS)	377
Das Launcher-Icon	378
Versionierung bedenken	378
Crashlytics und Analytics einbinden	379
Apps an Testpersonen verteilen	382
Firebase App Distribution	383
Eine .apk- oder lieber eine .aab-Datei?	385
Verteilung über den Google Play Store	385
App-Bundles signieren	386
Eine neue App im Google Play Store anlegen	387
Eine Testversion per Google Play Store erstellen und verteilen	388
Ein Production Release per Google Play Store erstellen und verteilen	390

Kapitel 23 Der iOS-Build 393

Voraussetzungen	393
Vorbereitung eines iOS-Builds	393
Apple-Developer-Account	394

22 Inhaltsverzeichnis

Registrieren Sie Ihre App	394
Bundle ID	395
App im App Store Connect anlegen	395
Erstellen Sie einen Build mit Xcode	395
Xcode-Einstellungen prüfen	395
Ein erster Build	396
Apps an Testpersonen verteilen	397
Firebase App Distribution	397
Testflight	398
Fertig! Release Build erstellen und veröffentlichen	399
Recap: Testen, builden und veröffentlichen	400
Bye bye	400

TEIL VII TOP-TEN-TEIL 401

Kapitel 24 Unsere 10 Lieblings-Widgets 403

Widget 1: Chip	403
Widget 2: Wrap	403
Widget 3: CupertinoDatePicker und showDatePicker	403
Widget 4: PageView	404
Widget 5: Table	404
Widget 6: Hero	404
Widget 7: AnimatedContainer	404
Widget 8: Semantics	405
Widget 9: SliverAppBar	405
Widget 10: CustomPaint	405

Kapitel 25 Unsere 10 Flutter-Tipps und -Tricks 407

Tipp 1: Wenn Sie einen komischen Fehler haben, den Sie nicht lösen können ..	407
Tipp 2: Wenn ein iOS-Build nicht hinhaut	407
Tipp 3: Konsistente Benennung von Dateien und Klassen	408
Tipp 4: Arbeiten Sie mit einem Linter	408
Tipp 5: Formatieren Sie Ihren Code mit einem Formatter	408
Tipp 6: Verwenden Sie den automatischen Fix-Command von Dart	409
Tipp 7: Updaten Sie Flutter und Ihre Dependencies regelmäßig	409
Tipp 8: Wann für oder gegen ein Package oder Plug-in entscheiden	410
Tipp 9: Separieren Sie einzelne Widgets in eigene Klassen und separate Dateien	410
Tipp 10: Schauen Sie bei einem Flutter-Meetup vorbei	410

Abbildungsverzeichnis 411

Stichwortverzeichnis 417