

Inhaltsübersicht

I	Einführung	1
1	Schnellstart: unser erstes Angular-Projekt	3
2	Benötigte Werkzeuge: Editor, Node.js und Co.	11
3	Angular CLI: der Codegenerator für unser Projekt	19
II	TypeScript	25
4	Einführung in TypeScript	27
III	BookMonkey 5: Schritt für Schritt zur App	51
5	Projektvorstellung und Einrichtung	53
6	Komponenten: die Grundbausteine der Anwendung	73
7	Property Bindings: mit Komponenten kommunizieren	107
8	Event Bindings: Ereignisse in Komponenten verarbeiten ...	123
9	Power Tipp: Codeformatierung mit Prettier	137
10	NgModule: die Anwendung modularisieren	141
11	Dependency Injection: Code in Services auslagern	157
12	Routing: durch die Anwendung navigieren	175
13	Power Tipp: Chrome Developer Tools	213
14	HTTP-Kommunikation: ein Server-Backend anbinden	225
15	Reaktive Programmierung mit RxJS	243

16	Interceptors: HTTP-Requests abfangen und transformieren	301
17	Power Tipp: Analyse und Debugging mit den Angular DevTools	321
18	Formulare mit Template-Driven Forms	325
19	Formulare mit Reactive Forms	345
20	Formularvalidierung: die Eingaben prüfen	387
21	Pipes: Daten im Template formatieren	413
22	Direktiven: das Vokabular von HTML erweitern	433
23	Lazy Loading: Angular-Module asynchron laden	459
24	Guards: Routen absichern	471
25	Standalone Components: Komponenten ohne Module	485
 IV Projektübergreifende Themen		507
26	Qualität fördern mit Softwaretests	509
27	Barrierefreiheit (a11y)	577
28	Lokalisierung (l10n)	599
29	Internationalisierung (i18n)	605
 V Deployment: das Projekt ausliefern		629
30	Build und Deployment mit der Angular CLI	631
31	Angular-Anwendungen mit Docker bereitstellen	657

VI Fortgeschrittene Themen 677

- 32 State Management mit Redux und NgRx 679**
- 33 Server-Side Rendering mit Angular Universal 741**
- 34 Progressive Web Apps (PWA) 761**
- 35 Fortgeschrittene Konzepte der Angular CLI 783**

VII Wissenswertes 791

- 36 Fortgeschrittene Konzepte für Komponenten 793**
- 37 Weitere Features des Routers 819**
- 38 Nützliche Werkzeuge 827**
- 39 Web Components mit Angular Elements 835**
- 40 Angular Material und weitere UI-Komponenten-
sammlungen 843**
- 41 Angular updaten 847**
- 42 Klassen-Propertys in JavaScript und TypeScript 851**

VIII Anhang 857

- A Befehle der Angular CLI 859**
- B Operatoren von RxJS 867**
- C Matchers von Jasmine 871**
- D Abkürzungsverzeichnis 875**
- E Linkliste 877**
- Index 887**
- Nachwort 897**

Inhaltsverzeichnis

Vorwort	xxi
----------------------	------------

Aktualisierungen in der vierten Auflage	xxix
--	-------------

I Einführung 1

1 Schnellstart: unser erstes Angular-Projekt	3
2 Benötigte Werkzeuge: Editor, Node.js und Co.	11
2.1 Konsole, Terminal und Shell	11
2.2 Visual Studio Code	11
2.3 Google Chrome	14
2.4 Paketverwaltung mit Node.js und NPM	14
2.5 Codebeispiele in diesem Buch	16
3 Angular CLI: der Codegenerator für unser Projekt	19
3.1 Das offizielle Tool für Angular	19
3.2 Installation	20
3.3 Die wichtigsten Befehle	21
3.4 Autovervollständigung einrichten	22

II TypeScript 25

4 Einführung in TypeScript	27
4.1 TypeScript einsetzen	27
4.2 Variablen: const, let und var	29
4.3 Die wichtigsten Basistypen	31
4.4 Klassen	34
4.5 Interfaces	38
4.6 Template-Strings	39
4.7 Arrow-Funktionen/Lambda-Ausdrücke	40
4.8 Spread-Operator und Rest-Syntax	41
4.9 Weitere Features von JavaScript und TypeScript	44
4.10 Konfiguration	48

III	BookMonkey 5: Schritt für Schritt zur App	51
5	Projektvorstellung und Einrichtung	53
5.1	Unser Projekt: BookMonkey	53
5.2	Projekt mit der Angular CLI initialisieren	58
5.3	Aufbau des neuen Projekts	59
5.4	Das Projekt starten	65
5.5	Globale Styles einbinden: book-monkey5-styles	66
5.6	Statische Codeanalyse mit ESLint	68
6	Komponenten: die Grundbausteine der Anwendung	73
6.1	Komponenten	73
6.2	Komponenten in der Anwendung verwenden	78
6.3	Komponenten generieren mit der Angular CLI	79
6.4	Umgang mit Propertys von Komponenten	80
6.5	Template-Syntax	82
6.6	Elemente gruppieren mit <ng-container>	89
6.7	Den BookMonkey erstellen: eine Buchliste anzeigen	94
7	Property Bindings: mit Komponenten kommunizieren ...	107
7.1	Komponenten verschachteln	107
7.2	Eingehender Datenfluss mit Property Bindings	108
7.3	Daten in Kindkomponenten verarbeiten	109
7.4	Property Bindings für native Elemente	110
7.5	Property Bindings notieren	111
7.6	Sonderformen von Property Bindings	113
7.7	Lifecycle Hooks von Komponenten	115
7.8	Den BookMonkey erweitern: Listeneinträge in eigener Komponente abbilden	117
8	Event Bindings: Ereignisse in Komponenten verarbeiten .	123
8.1	Native DOM-Events	124
8.2	Eigene Events definieren	126
8.3	Den BookMonkey erweitern: Buchdetails anzeigen	128
9	Power Tipp: Codeformatierung mit Prettier	137
10	NgModule: die Anwendung modularisieren	141
10.1	Module in Angular	141
10.2	Grundaufbau eines Moduls	142
10.3	Bestandteile eines Moduls deklarieren	143
10.4	Andere Module importieren	143
10.5	Bestandteile aus Modulen exportieren	145
10.6	Anwendung in Feature-Module aufteilen	146

10.7	Wiederverwendbarkeit: Shared Module	148
10.8	Den BookMonkey erweitern: die Anwendung modularisieren	149
11	Dependency Injection: Code in Services auslagern	157
11.1	Abhängigkeiten anfordern	159
11.2	Services in Angular	160
11.3	Abhängigkeiten registrieren	160
11.3.1	Abhängigkeiten explizit registrieren mit providers...	160
11.3.2	Tree-Shakable Providers mit providedIn	162
11.4	Abhängigkeiten ersetzen	163
11.5	Eigene Tokens definieren mit InjectionToken	166
11.6	Abhängigkeiten anfordern mit @Inject()	167
11.7	Abhängigkeiten anfordern mit inject()	168
11.8	Multiprovider: mehrere Abhängigkeiten im selben Token	169
11.9	Providers in Komponenten registrieren	169
11.10	Den BookMonkey erweitern: einen Service nutzen	170
12	Routing: durch die Anwendung navigieren	175
12.1	Routen konfigurieren	176
12.2	Router einbinden: das AppRoutingModuleModule	177
12.3	Routing in Feature-Modulen	178
12.4	Komponenten anzeigen	181
12.5	Root-Route	182
12.6	Weiterleitung auf eine andere Route	182
12.7	Wildcard-Route	183
12.8	Links setzen	183
12.9	Routenparameter	185
12.10	Verschachtelung von Routen	188
12.11	Aktive Links stylen	191
12.12	Route programmatisch wechseln	192
12.13	Seitentitel setzen	193
12.14	Pfade in Single-Page-Applikationen	196
12.15	Den BookMonkey erweitern: Routing integrieren	197
13	Power Tipp: Chrome Developer Tools	213
14	HTTP-Kommunikation: ein Server-Backend anbinden	225
14.1	Modul einbinden	226
14.2	Requests mit dem HttpClient durchführen	227
14.3	Optionen für den HttpClient	229
14.4	Ausblick: Codegenerierung mit OpenAPI	233
14.5	Den BookMonkey erweitern: Daten über HTTP abfragen	234

15	Reaktive Programmierung mit RxJS	243
15.1	Alles ist ein Datenstrom	243
15.2	Observables sind Funktionen	245
15.3	Das Observable aus RxJS	247
15.4	Observables abonnieren	249
15.5	Observables erzeugen	251
15.6	Observables und Promises	254
15.7	Operatoren: Datenströme modellieren	255
15.8	Heiße Observables, Multicasting und Subjects	259
15.9	Subscriptions verwalten & Memory Leaks vermeiden	266
15.10	Observables subscriben mit der AsyncPipe	270
15.11	Fehler behandeln	273
15.12	Flattening-Strategien für Higher-Order Observables	276
15.13	Den BookMonkey erweitern: Observables mit der AsyncPipe auflösen	282
15.14	Den BookMonkey erweitern: Typeahead-Suche	286
15.15	Den BookMonkey erweitern: Fehlerbehandlung	296
16	Interceptors: HTTP-Requests abfangen und transformieren	301
16.1	Funktionsweise der Interceptors	302
16.2	Interceptors anlegen	302
16.3	Den Request manipulieren	304
16.4	Die Response verarbeiten	304
16.5	Interceptors einbinden	305
16.6	Interceptors als Funktionen	307
16.7	OAuth 2 und OpenID Connect	307
16.8	Den BookMonkey erweitern: API-Aufrufe mit Credentials anreichern	310
17	Powertipp: Analyse und Debugging mit den Angular DevTools	321
18	Formulare mit Template-Driven Forms	325
18.1	Angulars Ansätze für Formulare	325
18.2	Template-Driven Forms einrichten	326
18.3	Datenmodell in der Komponente	327
18.4	Template mit Two-Way Binding und ngModel	327
18.5	Eingaben validieren	328
18.6	Formularzustand verarbeiten	329
18.7	Formular abschicken	331
18.8	Formular zurücksetzen	331
18.9	Den BookMonkey erweitern: Template-Driven Forms nutzen .	333

19	Formulare mit Reactive Forms	345
19.1	Modul einbinden	346
19.2	Formularmodell in der Komponente	346
19.3	Template mit dem Modell verknüpfen	352
19.4	Eingebaute Validatoren nutzen	355
19.5	Formularzustand verarbeiten	357
19.6	Formular abschicken	358
19.7	Formular zurücksetzen	359
19.8	Formularwerte setzen	360
19.9	FormBuilder verwenden	361
19.10	Änderungen überwachen	363
19.11	Den BookMonkey erweitern: Reactive Forms nutzen	364
19.12	Den BookMonkey erweitern: Autor*innen erfassen	370
19.13	Den BookMonkey erweitern: Bücher bearbeiten	374
19.14	Welcher Ansatz ist der richtige?	384
20	Formularvalidierung: die Eingaben prüfen	387
20.1	Validatoren für einzelne Formularfelder	387
20.2	Validatoren für Formulargruppen und -Arrays	391
20.3	Validatoren kombinieren	393
20.4	Asynchrone Validatoren	394
20.5	Mit Fehlern arbeiten	397
20.6	Den BookMonkey erweitern: Felder für Autor*innen validieren	398
20.7	Den BookMonkey erweitern: ISBN-Format validieren	400
20.8	Den BookMonkey erweitern: existierende ISBN prüfen	402
20.9	Den BookMonkey erweitern: Fehlermeldungen anzeigen	405
21	Pipes: Daten im Template formatieren	413
21.1	Pipes verwenden	413
21.2	Eingebaute Pipes für den sofortigen Einsatz	414
21.3	Eigene Pipes entwickeln	424
21.4	Pipes in TypeScript nutzen	427
21.5	Den BookMonkey erweitern: Datum formatieren mit der DatePipe	428
21.6	Den BookMonkey erweitern: ISBN formatieren	429
22	Direktiven: das Vokabular von HTML erweitern	433
22.1	Was sind Direktiven?	433
22.2	Eigene Direktiven entwickeln	434

22.3	Attributdirektiven: Verhalten von Elementen ändern	436
22.3.1	Host Binding: Eigenschaften schreiben	437
22.3.2	Host Listener: Events abonnieren	439
22.3.3	Direktzugriff auf das Element mit ElementRef	440
22.3.4	Komponenten und Direktiven anfordern	441
22.4	Komposition mit Host-Direktiven	443
22.5	Strukturdirektiven: Elemente hinzufügen und entfernen	445
22.6	Den BookMonkey erweitern: Löschen mit Dialog bestätigen ..	449
22.7	Den BookMonkey erweitern: die Löschfunktion absichern ...	453
23	Lazy Loading: Angular-Module asynchron laden	459
23.1	Warum Module asynchron laden?	459
23.2	Das technische Konzept	460
23.3	Lazy Loading verwenden	461
23.4	Module asynchron vorladen: Preloading	464
23.5	Den BookMonkey erweitern: Module asynchron nachladen ..	465
24	Guards: Routen absichern	471
24.1	Grundlagen zu Guards	471
24.2	Guards verwenden	472
24.3	Guards implementieren	473
24.4	Guards als Klassen	478
24.5	Guards und Sicherheit	479
24.6	Den BookMonkey erweitern: die Admin-Route absichern	480
25	Standalone Components: Komponenten ohne Module ...	485
25.1	NgModule und Standalone Components	485
25.2	Standalone Components erzeugen	487
25.3	Abhängigkeiten definieren	487
25.4	Standalone Components in NgModules nutzen	489
25.5	Routing	490
25.6	Anwendungen ohne Module: AppComponent direkt bootstrappen	493
25.7	Projektstruktur	497
25.8	Fazit	498
25.9	Den BookMonkey erweitern: Pipes und Direktiven standalone verwenden	499
25.10	Den BookMonkey erweitern: Bücher-Feature mit Standalone Components	502

IV Projektübergreifende Themen 507

26	Qualität fördern mit Softwaretests	509
26.1	Softwaretests	509
26.2	Vorgehen beim Testing	510
26.3	Test-Framework Jasmine	512
26.4	Test-Runner	517
26.4.1	Karma	517
26.4.2	Alternative: Jest	518
26.4.3	E2E-Test-Runner wählen	518
26.5	Unit- und Integrationstests mit Karma	519
26.5.1	TestBed: die Testbibliothek von Angular	519
26.5.2	Isolierte Unit-Tests: Services testen	521
26.5.3	Isolierte Unit-Tests: Pipes testen	523
26.5.4	Isolierte Unit-Tests: Komponenten testen	524
26.5.5	Shallow Component Test: einzelne Komponenten testen	528
26.5.6	Integrationstests: mehrere Komponenten testen	532
26.5.7	Abhängigkeiten durch Stubs ersetzen	534
26.5.8	Abhängigkeiten durch Mocks ersetzen	538
26.5.9	Leere Komponenten als Stubs oder Mocks einsetzen	541
26.5.10	HTTP-Requests testen	542
26.5.11	Komponenten mit Routen testen	546
26.5.12	Asynchronen Code testen	550
26.5.13	Code Coverage Report	556
26.5.14	Zusammenfassung: Tests mit Karma und Jasmine ...	557
26.6	Jest: ein alternativer Test-Runner mit zusätzlichen Features...	558
26.7	Oberflächentests mit Cypress	562
26.8	Component Tests mit Cypress: Komponenten isoliert testen..	572
27	Barrierefreiheit (a11y)	577
27.1	Gesetze und Standards	579
27.2	Features von Angular	584
27.3	ESLint-Regeln	588
27.4	Angular Component Development Kit (CDK)	589
27.5	Verifizierung & Tools zur Unterstützung	595
28	Lokalisierung (l10n)	599
28.1	Lokalisierung für ein spezifisches Locale	600
28.2	Mehrere Sprachdefinitionen laden	601
28.3	Pipes mit einem spezifischen Locale nutzen	602

29	Internationalisierung (i18n)	605
29.1	Was bedeutet Internationalisierung?	605
29.2	Der Übersetzungsprozess in Angular	606
29.3	Texte für die Übersetzung markieren und extrahieren	607
29.3.1	Projekt vorbereiten	607
29.3.2	Nachrichten im HTML mit dem i18n-Attribut markieren	608
29.3.3	Nachrichten im TypeScript-Code mit \$localize markieren	609
29.3.4	Feste IDs vergeben	610
29.3.5	Nachrichten extrahieren und übersetzen	611
29.4	Übersetzung während des Build-Prozesses	613
29.5	Übersetzung zur Laufzeit	619
29.6	Technische Einschränkungen	627
V	Deployment: das Projekt ausliefern	629
30	Build und Deployment mit der Angular CLI	631
30.1	Build konfigurieren (angular.json)	631
30.2	Build ausführen	635
30.3	Bundles	636
30.3.1	Weitere Bundles und Dateien	637
30.3.2	Budgets konfigurieren	638
30.3.3	Bundles analysieren mit source-map-explorer	639
30.4	Umgebungen konfigurieren	640
30.5	Ahead-of-Time-Kompilierung: die Templates umsetzen	646
30.6	Webserver konfigurieren und die Anwendung ausliefern	649
30.7	ng deploy: Deployment mit der Angular CLI	652
30.8	Ausblick: Deployment mit einem Build-Service	654
31	Angular-Anwendungen mit Docker bereitstellen	657
31.1	Docker	658
31.2	Docker Registry	659
31.3	Lösungsskizze	659
31.4	Eine Angular-App über Docker bereitstellen	660
31.5	Build Once, Run Anywhere: Konfiguration über Docker verwalten	664
31.6	Multi-Stage Builds	670
31.7	Grenzen der vorgestellten Lösung	674
31.8	Fazit	675

VI Fortgeschrittene Themen 677

32	State Management mit Redux und NgRx	679
32.1	Ein Modell für zentrales State Management	680
32.2	Das Architekturmodell Redux	691
32.3	NgRx: Reactive Extensions for Angular	693
32.3.1	Projekt vorbereiten	694
32.3.2	Store einrichten	694
32.3.3	Schematics nutzen	694
32.3.4	Grundstruktur	695
32.3.5	Feature anlegen	696
32.3.6	Struktur des Feature-States definieren	698
32.3.7	Actions: Kommunikation mit dem Store	699
32.3.8	Dispatch: Actions in den Store senden	701
32.3.9	Reducers: den State aktualisieren	702
32.3.10	Selektoren: Daten aus dem State lesen	706
32.3.11	Effects: Seiteneffekte ausführen	711
32.4	Debugging mit den Redux DevTools	717
32.5	Redux und NgRx: Wie geht's weiter?	720
32.5.1	Actions gruppieren mit <code>createActionGroup()</code>	720
32.5.2	Routing	721
32.5.3	Entity Management	721
32.5.4	Testing	724
32.5.5	Hilfsmittel für Komponenten: <code>@ngrx/component</code>	733
32.5.6	Facades: Zustandsverwaltung abstrahieren	735
32.6	Ausblick: lokaler State mit <code>@ngrx/component-store</code>	738
33	Server-Side Rendering mit Angular Universal	741
33.1	Single-Page-Anwendungen, Suchmaschinen und Start-Performance	742
33.2	Dynamisches Server-Side Rendering	745
33.3	Statisches Pre-Rendering	750
33.4	Hinter den Kulissen von Angular Universal	753
33.5	Browser oder Server? Die Plattform bestimmen	754
33.6	Routen ausschließen	755
33.7	Wann setze ich serverseitiges Rendering ein?	757
33.8	Ausblick: Pre-Rendering mit Scully	758
34	Progressive Web Apps (PWA)	761
34.1	Die Charakteristiken einer PWA	761
34.2	Service Worker	762
34.3	Eine bestehende Angular-Anwendung in eine PWA verwandeln	763

34.4	Add to Homescreen	765
34.5	Offline-Funktionalität	768
34.6	Push-Benachrichtigungen	773

35 Fortgeschrittene Konzepte der Angular CLI 783

35.1	Workspace und Monorepo: Heimat für Apps und Bibliotheken	783
35.1.1	Applikationen: Angular-Apps im Workspace	784
35.1.2	Bibliotheken: Code zwischen Anwendungen teilen ..	786
35.2	Schematics: Codegenerierung mit der Angular CLI	788

VII Wissenswertes 791

36 Fortgeschrittene Konzepte für Komponenten 793

36.1	Else-Block für die Direktive ngIf	793
36.2	TrackBy-Funktion für die Direktive ngFor	794
36.3	Container und Presentational Components	796
36.4	Content Projection: Inhalt des Host-Elements verwenden	800
36.5	Lifecycle Hooks	802
36.6	Change Detection	805

37 Weitere Features des Routers 819

37.1	Auxiliary Routes: mehrere RouterOutlets verwenden	819
37.2	Erweiterte Konfigurationen für den Router	820
37.3	Resolvers: Daten beim Routing vorladen	822

38 Nützliche Werkzeuge 827

38.1	Monorepos mit Nrwl Nx	827
38.2	Angular-Anwendungen dokumentieren und visualisieren	830
38.2.1	Compodoc	831
38.2.2	Storybook	832

39 Web Components mit Angular Elements 835

40 Angular Material und weitere UI-Komponenten- sammlungen 843

41 Angular updaten 847

42 Klassen-Propertyts in JavaScript und TypeScript 851

VIII	Anhang	857
A	Befehle der Angular CLI	859
B	Operatoren von RxJS	867
C	Matchers von Jasmine	871
D	Abkürzungsverzeichnis	875
E	Linkliste	877
Index		887
Nachwort		897