Wolfgang Driebe

Android und Java

Android-Apps mit Java programmieren

1. Auflage

Bestellnummer 01130

Bildungsverlag EINS
westermann

Die in diesem Werk aufgeführten Internetadressen sind auf dem Stand zum Zeitpunkt der Drucklegung. Die ständige Aktualität der Adressen kann vonseiten des Verlages nicht gewährleistet werden. Darüber hinaus übernimmt der Verlag keine Verantwortung für die Inhalte dieser Seiten.

service@bv-1.de
www.bildungsverlag1.de

Bildungsverlag EINS GmbH
Ettore-Bugatti-Straße 6-14, 51149 Köln

ISBN 978-3-427-**01130**-9

westermann GRUPPE

Inhaltsverzeichnis

Vorwort

Android-Apps mit Java programmieren

Android finden Sie auf den meisten Smartphones als Betriebssystem. Marktuntersuchungen ergaben für 2016, dass Android auf mehr als 80 % der Smartphones weltweit, als Betriebssystem installiert ist. Grund für die weite Verbreitung ist die Offenheit von Android. Für Sie als Schülerin oder Schüler liegt der Vorteil der Offenheit in der kostenlosen Entwicklungsumgebung, das Android Studio, das Sie auf Ihrem Rechner mit Windows als Betriebssystem installieren können. Für Rechner mit *Linux* oder *Mac OS X* stehen ebenfalls kostenlose Entwicklungsumgebungen zum Download zur Auswahl.

Unterstützt werden Sie beim Programmieren auf zahlreichen deutsch- und englischsprachigen Foren im Internet. Hier finden Sie Antworten auf mögliche Fragen. Beim Entwickeln der Apps unterstützt Sie das Android Studio mit der Anzeige von Fehlern, Hinweisen und Vorschlägen für alternative Codierungen.

Java ist die Programmiersprache, mit der Sie Programme schreiben, die unter dem Betriebssystem Android laufen. Java ist auch eine der am meist genutzten Programmiersprachen in der Datenverarbeitung.

App-Programmierung wird Sie begeistern. Wenn die ersten programmierten Apps auf dem Smartphone laufen und Sie die Apps Ihren Lehrerinnen und Lehrern, Ihren Mitschülern und Mitschülerinnen, Ihren Freunden und Freundinnen vorführen, werden Sie dieses Buch nicht mehr aus der Hand legen. Sie werden es in kürzester Zeit, von Projekt zu Projekt und von Lernerfolgen begleitet, durcharbeiten. Die Beispiele und die Aufgaben sind App-Ideen, die Sie privat oder an Ihrem Arbeitsplatz weiter entwickeln können. Lassen Sie Ihre Kreativität und Ihr Fachwissen in die Weiterentwicklung der Ideen einfließen, um über kurz oder lang eigene Apps zu veröffentlichen.

Nach dem Studium des Buches haben Sie sich die Grundlagen der App-Programmierung erarbeitet, das heißt, Sie haben Grundlagen der Programmiersprache Java und Basiswissen über Android als Betriebssystem erworben.

Erstmalig arbeiten Sie gemeinsam mit den Lernmedien Smartphone, Rechner und Buch im Unterricht, um sich erfolgreich auf schulische Prüfungen und die Abschlussprüfungen in den Berufen der Informatik vorzubereiten.

Die Entwicklung von Apps für technische und wirtschaftliche Problemlösungen gehört zu den Herausforderungen, denen sich die Ausbildungsbetriebe der Datenverarbeitung täglich stellen. Mit dem Studium dieses Buches verbessern Sie Ihre Qualifikationen für das jetzige und spätere Berufsleben.

Das Buch

Die einzelnen Kapitel des Buches unterteilen sich in Projekte. Das Buch führt Sie als Lesenden mit jedem neuen Projekt tiefer in die Programmiersprache Java und die Android-Welt ein. Das Projekt „Carsharing Hamburg“ am Ende des Buches nutzt die Datenbank *SQLite* und wendet die Datenbanksprache *SQL* (*Structured Query Language*) in der Android Umgebung an.

Zu Beginn jedes Projekts wird aufgezählt, welche Fähigkeiten Sie während der Bearbeitung des Projekts erwerben und welche neuen Inhalte der Java und Android Welt Sie lernen. Die Entwicklung der Apps gliedert sich in die Phasen Gestaltung der Oberfläche (*xml*-Datei), Programmierung der Java-Datei und Test der Anwendung. Farbig hinterlegt unterscheiden sich Java-Texte, Android-Beschreibungen und notwendiges Hintergrundwissen.

Java

Android

Hintergrundwissen

Mit der Bearbeitung der Aufgaben am Ende jeden Kapitels üben und vertiefen Sie das Gelernte.

Sie entwickeln mit Java als Programmiersprache und Android als Betriebssystem Apps für Smartphone und Tablet. Entwicklungsumgebung ist das Android Studio. Sie programmieren die Apps auf Ihrem Rechner mit Windows oder *Linux* als Betriebssystem.

Die von Ihnen entwickelten Apps testen Sie auf dem eigenem Smartphone oder Tablet. Voraussetzung ist das Betriebssystem Android. Haben Sie kein Android Smartphone, testen Sie mit den installierten Smartphone Simulatoren. Alle Projekte dieses Buches wurden mit einem Smartphone-Emulator – *Android Virtual Device* (*AVD*) – auf einem Windows-Rechner getestet.

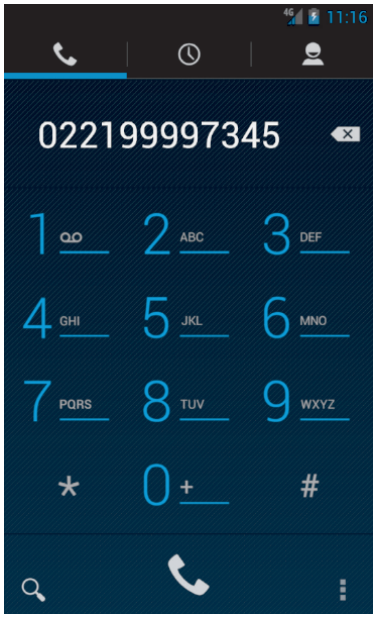
Inhaltliches wie z. B. *EditText*, *If*-Anweisung, Klasse und *query()* werden erst exemplarisch codiert und dann allgemein beschrieben. Beispiele verdeutlichen die theoretischen Ausführungen. Projekte und Aufgaben sind so angelegt, dass Sie erst mit einfachen und dann mit schwierigeren Problemen konfrontiert werden.

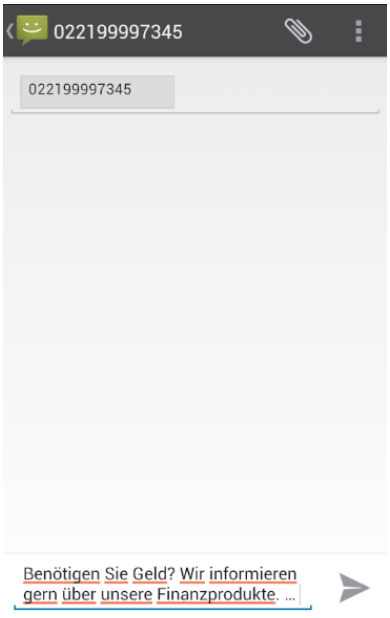
Das Einfügen von Programmtext wird Ihnen mit sich wiederholenden Codierungen in *xml* oder Java erleichtert. Diese Zeilen sind fett und kursiv formatiert.

xml-Zeile: ***android:gravity="right" />***
Java-Zeile: ***setContentView(R.layout.activity_main2;***

Zu den Online-Materialien unter *BuchPlusWeb* gehören die vielen Projekte und verwendeten Materialien. Die Lösungen der zahlreichen Aufgaben stehen den Lehrern und Lehrerinnen im Westermann Webshop als kostenpflichtiger Download zur Verfügung.

Anhang A beschreibt die Installation von Java und Android.
Anhang B erklärt, wie Sie mit dem eigenen Smartphone oder Tablet Apps entwickeln und testen.
Anhang C führt aus, wie Sie Ihre Apps mit einem Start-Icon und als *APK*-Datei verteilen.
Anhang D zeigt den Rahmen eines RelativLayouts und eines *LinearLayouts* – den Layouts, mit denen Sie in dem Buch arbeiten.





3.2 Berliner Radverleih

Realisieren Sie eine App, die einen Berliner Fahrradverleih simuliert.

Tagespreise des Berliner Radverleihs:

Citybike mit 3-Gang-Nabenschaltung:	15,00 €
E-Bike mit 3-Gang-Nabenschaltung und 400 Wattstunden-Akku:	25,00 €

Zusatzausstattung:

5-Gang-**Nabenschaltung**:	1,00 €
7-Gang-**Nabenschaltung**:	2,00 €
Satteltasche(n) nur Citybike jeweils	4,00 €

E-Bike:

500 **Wattstunden**-Akku:	2,00 €
600 **Wattstunden**-Akku:	4,00 €
Navi:	5,00 €

Während der Entwicklung des Projekts „Berliner Radverleih“

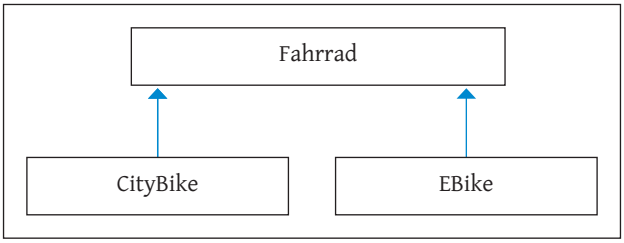
- programmieren Sie eigene Ober- und Unterklassen,
- erwerben Sie Wissen über *protected*- und *package*- Elemente,

- lernen Sie die Bedeutung des Schlüsselworts *super* kennen,
- erarbeiten Sie Kenntnisse über Polymorphie in Java,
- codieren Sie Konstante mit dem Schlüsselwort *final,*
- arbeiten Sie mit *spinner* und *checkbox,*
- entwickeln Sie Datumsdialoge,
- zeigen Sie programmatisch eine Adresse in *Maps* an,
- codieren Sie den Aufruf einer Internetseite.

Legen Sie das Projekt „Berliner Radverleih“ im Android Studio an.

- Programmieren Sie die Klasse „Fahrrad“ und die von dieser Klasse abgeleiteten Klassen „CityBike“ und „EBike“. Für jede der drei Dateien richten Sie, wie zu Beginn des Projekt „Bankkunden“ beschrieben, eine Java-Datei ein.

Hierarchie der eigenen Klassen



Die Oberklasse „Fahrrad“ enthält mit dem Modellnamen und der Anzahl der Gänge die gemeinsamen Attribute der Klassen „CityBike“ und „EBike“. Zu den Methoden der Elternklasse gehören *Konstruktor*, *Setter* und *Getter*, eine *zeigeFahrad-Funktion* und Funktionen zur Berechnung der Tages- und der Gesamtkosten. Für die Preise der Zusatzausstattung definieren Sie Konstante.

```
public class Fahrrad {
       final protected float PREIS_FUENF_GANG_SCHALTUNG  = 1.0f;
       final protected float PREIS_SIEBEN_GANG_SCHALTUNG = 2.0f;

       protected String modell;
       protected int gaenge;

       public Fahrrad(String modell, int gaenge) {
              this.modell = modell;
              this.gaenge = gaenge;
       }

       public String getModell() {
              return modell;
       }
       public void setModell(String modell) {
              this.modell = modell;
       }
```

```
        public int getGaenge() {
                return gaenge;
        }

        public void setGaenge(int gaenge) {
                this.gaenge = gaenge;
        }

        public String zeigeFahrrad() {
                return (modell + ", " + gaenge + " Gang-Schaltung, ");
        }

        public float radKostenProTag() {

                if (gaenge == 5)
                        return PREIS_FUENF_GANG_SCHALTUNG;
                else
                        if (gaenge == 7)
                        return PREIS_SIEBEN_GANG_SCHALTUNG;
                  else
                        return 0.0f;
        }

        public float radKostenGesamt(int tage) {

                float kosten = radKostenProTag() * tage;
                return (kosten);
        }
}
```

Codieren Sie die Klasse „CityBike“ als Unterklasse der Klasse „Fahrrad“.

Für den Tagespreis des *City-Bikes* und den Preis einer Satteltasche bestimmen Sie Konstante. Als zusätzliches Attribut deklarieren Sie in dieser Klasse eine Variable für die Anzahl der Taschen. Zu der Anzahl der Taschen gehört jeweils eine *get-* und eine *set-Funktion*. Der Konstruktor, die Methoden *zeigeFahrrad(), radKostenProTag()* und *radKostenGesamt(...)* verweisen mit dem Schlüsselwort *super* auf Attribute, Ausgaben und Berechnungen in der Oberklasse.

```
public class CityBike extends Fahrrad{

        final protected float PREIS_CITYBIKE = 15.0f;
        final protected float PREIS_TASCHE = 4.0f;

        protected int anzahlSatteltaschen;

        public CityBike(String modell, int gaenge, int anzahlSattel-
        taschen) {
                super(modell, gaenge);
                this.anzahlSatteltaschen = anzahlSatteltaschen;
        }
        public int getAnzahlSatteltaschen() {
                return anzahlSatteltaschen;
        }
```

gesamtKosten ()

```
private void gesamtKosten() {

    switch (radTyp) {
        case 1:
            ausgabeGesamtPreis.setText(citybike.radKostenGesamt(tage)
                                                              + " €");
            break;
        case 2:
            ausgabeGesamtPreis.setText(ebike.radKostenGesamt(tage)
                                                              + " €");
            break;
    }
}
```

Mit *switch* untersuchen Sie die Variable *radtyp* und rufen die überschriebene Methode *radKostenGesamt(...)* entweder mit der Referenz auf ein *citybike* oder *ebike* auf.

Parameter dieser Funktion sind die Tage und der Rückgabewert; der Gesamtpreis, wird von *ausgabeGesamtPreis.setText(...)* angezeigt.

findeVerleih()

```
private void findeVerleih() {
        Uri karteUri = Uri.parse("geo:0,0?" + "q=Georgenstraße%
          2014%2010117%20Berlin");
        // Uri karteUri = Uri.parse("geo:52.520188,13.393166?z=17");
        Intent karteIntent = new Intent(Intent.ACTION_VIEW);
        karteIntent.setData(karteUri);

        startActivity(karteIntent);
}
```

Adresse *anzeigen* (Maps)

Sie initialisieren ein Objekt der Klasse *Uri* mit der Methode *parse(...)*

Uri uriName = Uri.parse(„geo:0,0?“ + “q=Adresse“)

Leerzeichen werden bei der Adresseingabe durch %20 ersetzt. Für die Adresse des Bahnhofs Berlin-Friedrichstraße in der Georgenstraße 14, 10117 Berlin, in dessen Nähe Sie Fahrradverleihe finden, notieren Sie als Adresse

Georgenstraße%20400%2010117%20Berlin

Alternativ decodieren Sie eine Adresse im gängigen Adressformat.

String adresse = „Georgenstraße 14, 10117 Berlin“
Uri uri = Uri.parse(„geo:0,0?“ + „q=“ + Uri.encode(„adresse „))

Das Exemplar der Klasse *Intent* mit der *Aktion ACTION_VIEW* erzeugen Sie über

Intent intentName = new Intent(Intent.ACTION_VIEW).

Mit **intentName.setData(uriName)**

weisen Sie dem *Intent* die Adresse zu und

startActivity(intentName)

führt zum Start der Anwendung *Maps*.

Es ist möglich, die Koordinaten 0,0 des geo-Schemas durch Breiten-(52.520188) und Längengrad (13.393166) des Bahnhofs Berlin- Friedrichstraße zu ersetzen. z = 17 steht für den Zoomfaktor 17.

Uri uriName = Uri.parse(„geo:Breitengrad,Längengrad?Zoomfaktor“)

Beispiel 1

```
Uri uri = Uri.parse(„geo:0,0?“ + „q= Platz%20der%20Deutschen%20Einheit%201%
2020457%20Hamburg“);

Intent intent = new Intent(Intent.ACTION_VIEW);
intent.setData(uri);
startActivity(intent);
```

Der Standort der Elbphilharmonie in Hamburg wird angezeigt.

Beispiel 2

```
Uri uri = Uri.parse(„geo: 48.1310640, 11.5844570?z=17“);

Intent intent = new Intent(Intent.ACTION_VIEW);
intent.setData(uri);
startActivity(intent);
```

Führt zum Standort des Deutschen Museums in München.

Beispiel 3

```
Uri uri = Uri.parse(„geo:0,0?“ + „q=“ + Uri.encode(„Römerberg 27, 60311 Frankfurt“));

Intent karteIntent = new Intent(Intent.ACTION_VIEW);
karteIntent.setData(uri);
startActivity(karteIntent);
```

Der Standort des Rathauses in Frankfurt, des Römer, wird präsentiert.

zieleInBerlin()

```
private void zieleInBerlin() {
        Uri browserUri = Uri.parse("http://www.berlin.de/
          sehenswuerdigkeiten");
        Intent browserIntent = new Intent (Intent.ACTION_VIEW,
          browserUri);
        startActivity(browserIntent);
}
```

Webseite aufrufen

Sie generieren mit der Funktion *parse(...)* ein Objekt der Klasse *Uri*. Parameter ist die Internetadresse der Webseite.

Uri uriName = Uri.parse(„Internetadresse“)

Den *implizierten Intent* referenzieren Sie über den Konstruktor, mit Parametern für *Aktion* und *Uri*.

Intent = new Intent (Intent.ACTION_VIEW, uriName)

Auf den Start des *Intents*

startActivity (intentName)

reagiert der Standard Browser des Smartphones oder Tablets.

Beispiel

Uri uri = Uri.parse(„http://www.hnf.de“);
Intent intent = new Intent (Intent.ACTION_VIEW, uri);
startActivity(intent);

Anzeige der Homepage des Heinz Nixdorf MuseumsForum (Computermuseum) in Paderborn.

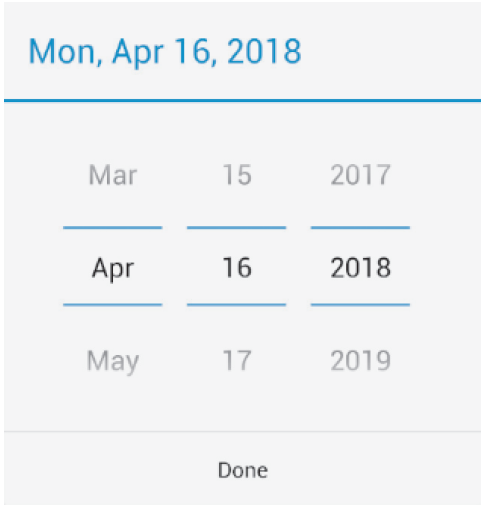
Über die Funktion *onClick(...)* des *onClickListeners* und die *if/else-Anweisung* steuern Sie den Aufruf der Methoden. Abschließend übergeben Sie den *Button* die *Listener*.

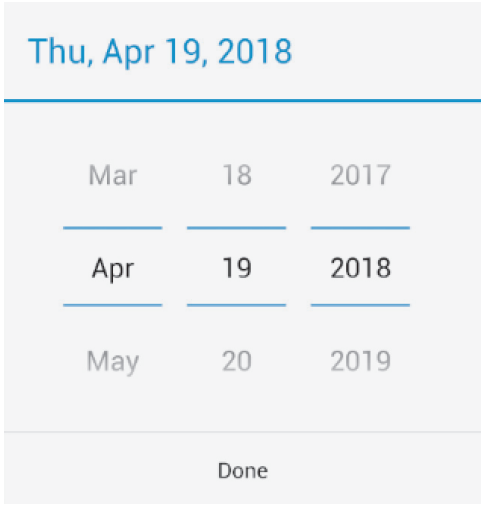
public void onClick(View v) {

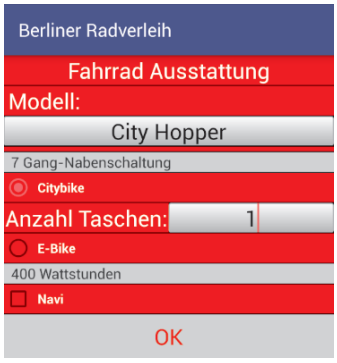
```
if (v == ausstattungFestlegen)
        startActivityForResult(intent, 88);
else
        if (v == beginnButton)
                ausleiheBeginn();
        else
                if (v == endeButton)
                        ausleiheEnde();
```

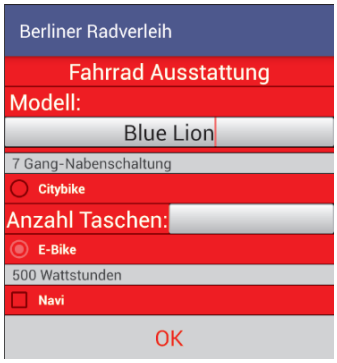
```
                        else
                                if (v == tageButton) {
                                        tageBerechnen();
                                        gesamtKosten();
                                }
                                else
                                        if(v == findenButton)
                                                findeVerleih();
                                else
                                        if(v == zieleButton)
                                                zieleInBerlin();
        }
};
ausstattungFestlegen.setOnClickListener(listener);
beginnButton.setOnClickListener(listener);
endeButton.setOnClickListener(listener);
tageButton.setOnClickListener(listener);
findenButton.setOnClickListener(listener);
zieleButton.setOnClickListener(listener);
```

Testen Sie das Projekt „Berliner Radverleih“ mit Emulator oder Smartphone.









3.3 Reisen und Verkehr

Implementieren Sie eine App, die den CO_2-Ausstoß und die Kosten beim Reisen mit unterschiedlichen Verkehrsmitteln berechnet.

CO_2 und Kosten verschiedener Transportmittel – durchschnittliche Werte

Auto	**CO_2:**	142,3 g/Pkm	-	**Kosten:**	0,56 Euro/km	**excl.** USt
Bahn	**CO_2:**	45,2 g/Pkm	-	**Kosten:**	0,17 Euro/km	**inkl.** USt
Flugzeug	**CO_2:**	230,7 g/Pkm	-	**Kosten:**	0,2 Euro/km	**inkl.** USt
Reisebus	**CO_2:**	30,3 g/Pkm	-	**Kosten:**	0,08 Euro/km	**inkl.** USt

(g/Pkm: Gramm pro Personenkilometer; excl./inkl. USt: ohne/mit Umsatzsteuer von 19 %)

Während der Entwicklung des Projekts „Reisen und Verkehr“

- arbeiten Sie mit abstrakten Klassen,
- programmieren und realisieren Sie eigene Schnittstellen (*Interface*),
- codieren Sie innere Klassen, anonyme Klassen und anonyme Objekte,

Bildquellenverzeichnis

Umschlag: fotolia.com, New York: links (adempercem), rechts (Lucky Dragon)