Hochschule Flensburg

Information und Kommunikation

Labor: Mobile Computing

Zeitraum: WS 2019/20

Name: Bluhm
Vorname: Johannes
Matrikelnummer: 670602

Semester: B-MI1

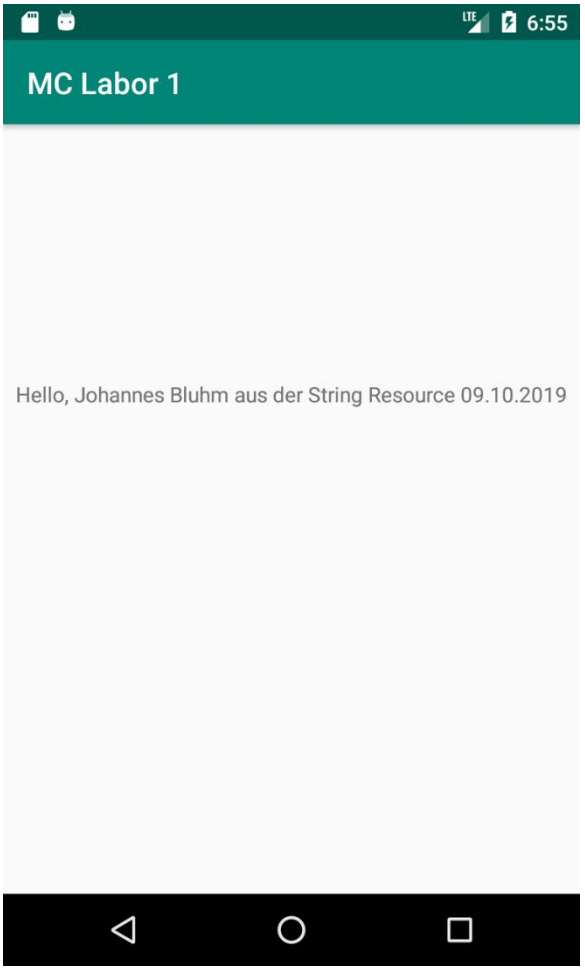
Inhaltsverzeichnis

Hello Android

Aufgabenstellung

Erstelle ein leeres Android Projekt, erstelle zuerst per Java Code, dann per XML Layout mittels Android View eine einfache String Ausgabe. Anschließend lagere den String in die strings.xml aus.

Ausgabe



Quellcode

MainActivity.java

```
package de.hsf.bljo;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        //TextView tv = new TextView(this);            //TextView
Objekt erstellen
        //tv.setText("Hello Android");                 //Text für die
neue Ausgabe
        //setContentView(tv);                          //TextObjekt
als UI Inhalt setzen
    }
}
```

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/varName"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.345" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Strings.xml

```
<resources>
    <string name="app_name">MC Labor 1</string>
    <string name="varName">Hello, Johannes Bluhm aus der String
Resource 09.10.2019</string>
</resources>
```

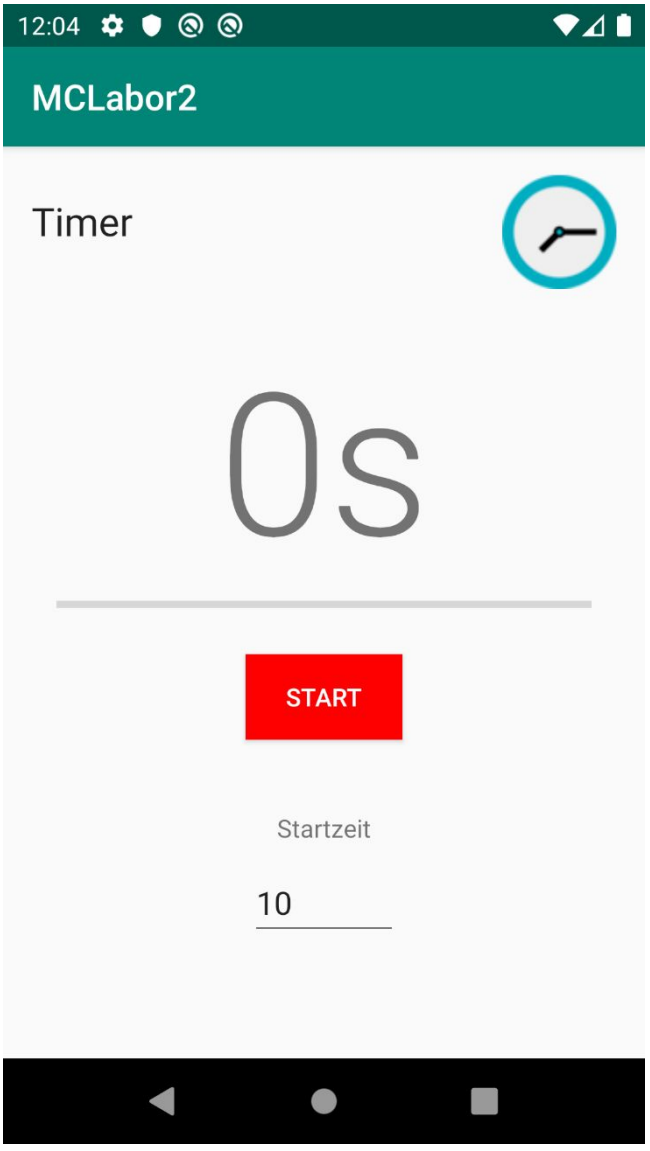
Labor 2

Responsive Design

Aufgabenstellung

*Ziel dieser Laboraufgabe: In dieser Aufgabe sollen Sie den Umgang mit Android GUI-Elementen üben und ein einfaches Responsive Layout gestalten. Verwendete Android-Techniken: 🎬 Klassen: ConstraintLayout, TextView, EditText, Button, ProgressBar 🎬 Initislisierung (View)-Objektreferenzen 🎬 Setzen von View-Attributen 🎬 Listener für Button Voraussetzung: Sie haben sich intensiv mit dem Foliensatz Responsive UI Design auseinandergesetzt. Laboraufgabe: Entwerfen Sie eine Android-App, mit einem responsive UI. Dieses UI soll auch für die nachfolgende Laboraufgabe GUI-Timer (Countdown Timer fürs Eierkochen) Verwendung finden. In diesem ersten Labor entwerfen Sie die Oberfläche hierfür mit erstmal minimaler Programmlogik. Die Oberfläche besteht aus folgenden Views (Widgets): 🎬 Logo (ImageView) 🎬 Erklärung Aufgabe dieser Activty :Timer (TextView) 🎬 Anzeigefeld: Restlaufzeit des Timers (TextView) 🎬 Fortschrittsanzeige (ProgressBar) 🎬 Start-/Stoppbutton (Button) 🎬 Optional: Divider (View) 🎬 Erklärung für das Eingabefeld (TextView) 🎬 Eingabe-/Anzeigefeld (EditText) Das Logo haben Sie mitgebracht (ca. 32*32, PNG-Format), dies wird oben rechts eingefügt. Der Benutzer soll über das Eingabefeld die Anzahl der Sekunden für die Countdown Zeit eingeben können (maximal 99s, also 2 Ziffern). Standardmäßig steht im Eingabefeld die Zahl „10“ für 10 Sekunden. Im Anzeigefeld steht standardmäßig „0“. Der Start-Stopp-Button ist mit „Start Timer“ beschriftet. Die Hintergrundfarbe des Buttons ist rot. Beispiel: Etwas Funktionalität soll doch vorhanden sein: Nach Drücken des Buttons: - wird der Inhalt des Eingabefeldes in das Anzeigefeld kopiert - ändert sich die Beschriftung des Buttons auf Stop Timer - wird der ‚Fortschritt‘ der ProgressBar auf den im Eingabefeld vorhandenen Wert gesetzt (z.B. Eingabefeld = 20, ProgressBar wird auf 20% des Maximalwertes gesetzt). Vorgaben für das Design: - Die App verwendet als Theme: Theme.AppCompat.NoActionBar - Es wird ein ConstraintLayout verwendet - Die Views sollen sinnvolle Größen haben, sodass sich ein ansprechendes UI ergibt. - Die Anzeige soll möglichst groß dargestellt werden. - Responsive: Die App ist auf Phones von 3,2“ – 6,3“ Zoll und Tabletts bis 10,1“ ohne Einschränkung bedienbar. Die Orientierung ist auf „Portrait“ festgelegt. Die Optik sollte auf die Phones optimiert sein. Für die Tabletts reicht eine bedienbare Variante. Es soll keine programmatische Lösung verwandt werden. Das responsive Verhalten soll alleinig mit dem Mitteln des ConstraintLayouts erreicht werden. - Schriftsatzgrößen werden soweit möglich über (vorhandene) Styles eingestellt. Da dieses UI wenige Elemente verwendet, können die Schriften etwas größer gewählt werden. - Schriftsatzgröße des Anzeigefeldes ist deutlich größer als die sonstigen Schriften. Fügen Sie in den Laborbericht Abbildungen Ihres Layouts von/auf folgenden Geräten ein (s. Foliensatz Responsive UI Design/Constraint Layout 2/Test von Layouts): - Phone 3,2“ - Phone 4“ - Phone 6“ - Tablett 10,1“ Achtung: Alle Vorgaben aus der Vorlesung bzgl. Namensgebung und Debugging müssen (sinnvoll) implementiert werden. Erzeugen sie für diese Aufgabe ein neues Projekt. Denken sie an die LogCat-Ausgaben die zwingend in jeder Methode mindestens einmal, wenn sinnvoll auch öfter, angewendet werden sollen. Falls Sie die Aufgabe nicht gefordert hat: - Gestalten Sie Ihr Constraint Layout so, dass es auch in Landscape-Orientierung auf den obigen Geräten funktioniert. - Für große Geräte werden größere Schriften verwandt (zumindest für die Anzeige) (einmalige ‚Umschaltung‘, nicht in mehreren Stufen anpassen, nicht programmatisch lösen)*

Ausgabe



Quellcode

MainActivity.java

```
package de.hsf.mclabor2;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        final Button startButton = findViewById(R.id.startButton);
        final EditText inputTime = (EditText)findViewById(R.id.inputTime);
        final TextView remainingTime =
(TextView)findViewById(R.id.remainingTime);
        final ProgressBar progressBar = (ProgressBar)
findViewById(R.id.progressBar);

        startButton.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                // Code here executes on main thread after user presses
button
                 Log.d("startButton pressed","");

                String time = inputTime.getText().toString();
                remainingTime.setText(time.toString() + "s");
                progressBar.setProgress(Integer.parseInt(time));
            }
        });
    }
}
```

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/hintTitle"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginLeft="16dp"
        android:layout_marginTop="27dp"
        android:text="Timer"
        android:textAppearance="@style/TextAppearance.AppCompat.Large"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <ImageView
        android:id="@+id/logo"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="16dp"
        android:layout_marginRight="16dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@drawable/clock_icon" />

    <TextView
        android:id="@+id/remainingTime"
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="100dp"
        android:text="0s"
        android:textAppearance="@style/TextAppearance.AppCompat.Display4"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <ProgressBar
        android:id="@+id/progressBar"
        style="?android:attr/progressBarStyleHorizontal"
        android:layout_width="300dp"
        android:layout_height="wrap_content"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/remainingTime" />

    <Button
        android:id="@+id/startButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:background="#FF0000"
        android:text="Start"
        android:textColor="#FFFFFF"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/progressBar" />

    <TextView
        android:id="@+id/hintInput"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="40dp"
        android:text="Startzeit"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/startButton" />

    <EditText
        android:id="@+id/inputTime"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:ems="4"
        android:inputType="number"
        android:maxLength="2"
        android:text="10"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/hintInput" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Labor 3

GUI Timer

Aufgabenstellung

Ziel dieser Laboraufgabe: In dieser Aufgabe sollen Sie den Umgang mit Android GUI-Elementen in einer einfachen timergesteuerten Anwendung üben. Verwendete Android-Techniken: 🎬 Klassen: TextView, EditText, Button, CountDownTimer, ProgressBar, MediaPlayer - SoundPool (o. Ä.) 🎬 Listener für Timer und Button Laboraufgabe: Entwerfen Sie eine Android-App, die einen Countdown-Timer (z.B. für Eierkochen, Mittagspause, Ende der Vorlesung …) realisiert. Die Oberfläche besteht aus folgenden Elementen (Widgets): 🎬 Eingabe-/Anzeigefeld (EditText) 🎬 Start-/Stoppbutton (Button) 🎬 Anzeigefeld: „Restlaufzeit des Timers“ - Restlaufzeit 🎬 Fortschrittsanzeige (ProgressBar) Der Benutzer soll über das Eingabe-/Anzeigefeld die Anzahl der Sekunden für die Countdown Zeit eingeben können. Standardmäßig steht im Eingabe/Anzeigefeld die Zahl „10“ für 10 Sekunden. Der Start-Stopp-Button ist mit „Start Timer“ beschriftet. Nach Drücken des Buttons ändert sich die Beschriftung des Buttons auf „Stop Timer“, das Herunterzählen startet. Unter dem Button erscheint die Fortschrittsanzeige (runterlaufend), darunter das Anzeigefeld „Restlaufzeit des Timers“. Hinter oder unter dem Text wird die Restlaufzeit des Timers angezeigt (die Zahlen werden dekrementiert). Die Fortschrittsanzeige verringert sich entsprechend. Drückt der Benutzer den Stopp-Button, wird der Countdown angehalten. Die Beschriftung des Buttons ändert sich wieder in „Start Timer“. Nach Betätigung des Buttons wird der Timer neu gestartet. Nach Ablauf der eingegebenen Zeit ertönt ein akustisches Signal. Die Beschriftung des Start-/Stopp-Buttons ändert sich in „Alarm aus“. Der Benutzer beendet den Alarm durch Drücken des Buttons. Die Beschriftung des Start-/Stopp-Buttons ändert sich in „Start Timer“. Zu jeder Zeit kann der Benutzer durch eine erneute Eingabe einer Countdownzeit den Ablauf, nachdem der Timer abgelaufen ist oder gestoppt wurde, durch Betätigen des Start-Buttons mit dem neuen Wert starten. Beschriften Sie alle Elemente ausreichend und selbsterklärend. Achtung: Alle Vorgaben aus der Vorlesung bzgl. Namensgebung und Debugging müssen (sinnvoll) implementiert werden. Erzeugen sie für diese Aufgabe ein neues Projekt. Denken sie an die LogCat-Ausgaben die zwingend in jeder Methode mindestens einmal, wenn sinnvoll auch öfters, angewendet werden sollen.

Ausgabe

Quellcode

MainActivity.java

```
package de.hsf.mclabor2;

import androidx.appcompat.app.AppCompatActivity;

import android.media.Ringtone;
import android.media.RingtoneManager;
import android.net.Uri;
import android.os.Bundle;
import android.os.CountDownTimer;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ProgressBar;
import android.widget.TextView;
```

```
public class MainActivity extends AppCompatActivity {

    boolean timerStarted = false;
    boolean timerDone = false;
    CountDownTimer myTimer;
    private Ringtone ringtone;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        final Button startButton = findViewById(R.id.startButton);
        final EditText inputTime = (EditText)findViewById(R.id.inputTime);
        final TextView remainingTime =
(TextView)findViewById(R.id.remainingTime);
        final ProgressBar progressBar = (ProgressBar)
findViewById(R.id.progressBar);

        Uri notification =
RingtoneManager.getDefaultUri(RingtoneManager.TYPE_ALARM);
        ringtone = RingtoneManager.getRingtone(getApplicationContext(),
notification);

        startButton.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                //Check if timer is already started
                if (!timerStarted) {
                    //Timer is not started, start it now

                     // Code here executes on main thread after user presses
button
                     Log.d("startButton pressed", "");

                    startButton.setText("Stop");
                    timerStarted = true;

                    String time = inputTime.getText().toString();
                    int timeInt = Integer.parseInt(time);
                    //remainingTime.setText(time.toString() + "s");
                     progressBar.setProgress(timeInt);
                    progressBar.setMax(timeInt);

                    // einfaches Timerobjekt z.B. in OnCreate()
instanzieren
                     myTimer = new CountDownTimer(timeInt * 1000, 100) {
                        public void onTick(long millisUntilFinished) {
                            // Logging: man beachte: verbose wg.
Wiederholung
                             Log.d("dbg", "CountDownTimer.onTick():
sUntilFinished: " + millisUntilFinished);

                            long timeRemaining = millisUntilFinished / 1000;
                            int timeRemainingInt = (int) timeRemaining;

                            remainingTime.setText(timeRemainingInt + 1 +
"s");
                            progressBar.setProgress(timeRemainingInt + 1);
                        }
```

```
                    public void onFinish() {
                        Log.d("dbg", "CountDownTimer.onFinish()");

                        remainingTime.setText("0s");
                        progressBar.setProgress(0);

                        startButton.setText("Ausschalten");
                        timerDone = true;

                        //Play ringtone
                        ringtone.play();
                    }
                };

                myTimer.start();
            } else if (timerStarted && !timerDone) {
                //Timer is started but not done
                Log.d("dbg", "timer has been stopped");
                myTimer.cancel();
                timerStarted = false;
                startButton.setText("Start");

            } else if (timerDone) {
                //Timer is done, reset it
                Log.d("dbg", "timer is done");
                ringtone.stop();
                //Reset sound
                startButton.setText("Start");
                timerStarted = false;
                timerDone = false;
            }
        }
    });
  }
}
```

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/hintTitle"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginLeft="16dp"
        android:layout_marginTop="27dp"
        android:text="Timer"
        android:textAppearance="@style/TextAppearance.AppCompat.Large"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
```

```
<ImageView
    android:id="@+id/logo"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="16dp"
    android:layout_marginEnd="16dp"
    android:layout_marginRight="16dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:srcCompat="@drawable/clock_icon" />

<TextView
    android:id="@+id/remainingTime"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="100dp"
    android:text="0s"
    android:textAppearance="@style/TextAppearance.AppCompat.Display4"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<ProgressBar
    android:id="@+id/progressBar"
    style="?android:attr/progressBarStyleHorizontal"
    android:layout_width="300dp"
    android:layout_height="wrap_content"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/remainingTime" />

<Button
    android:id="@+id/startButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:background="#FF0000"
    android:text="Start"
    android:textColor="#FFFFFF"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/progressBar" />

<TextView
    android:id="@+id/hintInput"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="40dp"
    android:text="Startzeit"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/startButton" />

<EditText
    android:id="@+id/inputTime"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="10dp"
    android:ems="4"
    android:inputType="number"
```

```
            android:maxLength="2"
            android:text="10"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toBottomOf="@+id/hintInput" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Unter Nexus 4 API 23 muss der Timer alle 100 ms aufgerufen werden, da wenn er alle 1000 ms aufgerufen, der letzte Tick nicht ausgeführt wird. Unter Nexus 5 API 29 tritt das Problem nicht auf.

Labor 4

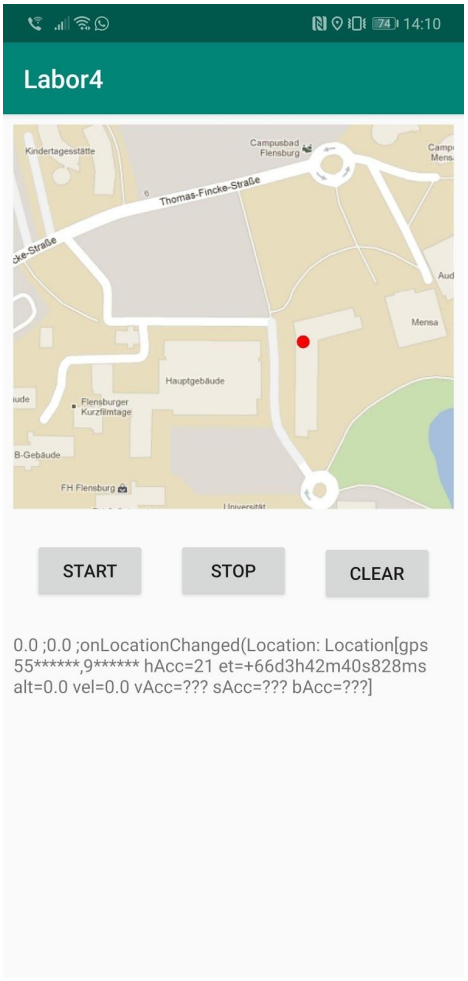
Location Based Services

Aufgabenstellung

Ziel dieser Laboraufgabe: In dieser Aufgabe entwickeln sie einen GPS-Tracer. Mit dem GPS Empfänger werden Ortskoordinaten bestimmt und dann in eine Karte eingezeichnet. Das Zeichnen in die Karte wird in einer eigenen, von Ihnen entworfenen Klasse realisiert. Die Karte wird aus einer Ressource geladen. Sie entwickeln eine App, in der die aktuelle Position in die Karte eingetragen wird. Verwendete Android-Techniken (u.a.): 🎬 Klassen: LocationManager, LocationListener, View, ... 🎬 Div. Listener 🎬 einbinden einer eigenen View-Klasse 🎬 erstellen/zeichnen einer elementaren Grafik 🎬 Zugriff vom View auf die „eigene" Activity (Daten holen, Observer) 🎬 Einbinden/Nutzen von Ressourcen (Grafik) 🎬 Umgang mit dem GPS-Emulator Hinweis: Erzeugen Sie ein neues Projekt, bei dem Minimum SDK auf „15" gesetzt ist. Laboraufgabe Location Based Services: Die Oberfläche besteht aus folgenden Elementen: 🎬 in der oberen Hälfte des Bildschirms wird die Karte angezeigt (Ressource fhfl_map) 🎬 darunter eine Zeile mit drei Buttons: Start, Stop und Clear 🎬 der restliche Platz unten wird von einem Textfeld belegt, in dem Debuginformationen (im wesentlichen GPS-Infos) ausgegeben werden (kleine gerade noch lesbare Schriftgröße) Nach dem Betätigen des Start-Buttons wird der GPS-Empfänger aktiviert und im Debug-Textfeld werden GPS-Statusinformationen ausgegeben. Sobald gültige Koordinaten empfangen werden, werden diese in die Karte eingezeichnet (jeweils nur die aktuelle Position). Betätigen des StopButtons deaktiviert den GPS-Empfänger. Betätigen des Clear-Buttons löscht die eingezeichnete Position. Sollte vor dem Drücken des Start-Buttons noch eine Eintragung in der Karte vorhanden sein, so soll auch diese beim Drücken des Start-buttons gelöscht werden. Implementation Die folgenden Tips beschreiben das prinzipielle Vorgehen zum Entwerfen dieser App. Detaillösungen werden Ihnen überlassen. Kommunikationstechnologie 1 Labor Mobile Computing Sie erhalten die Datei fhfl_map.jpg, die eine Karte des FH-Geländes zeigt. Die linke obere Ecke der Karte „besitzt" die Koordinate 54.776627°N, 9.448113°E. Die rechte untere Ecke: 54.774028°N, 9.453253°E. Die Auflösung der Karte bestimmen Sie mit den in der Vorlesung diskutierten Routinen. Die Aufgabe gliedert sich in zwei Bereiche: Elementare Grafikprogrammierung und GPS-Handling. Beginnen Sie mit dem GPS Teil: 🎬 Anlegen der Oberfläche nach den obigen Vorgaben. Binden Sie noch nicht die Grafik (die Ressource) ein, sondern legen sie erst mal einfach ein View-Objekt an. 🎬 Im Start-Button-Listener GPS aktivieren implementieren 🎬 Im Stop-Button-Listener GPS deaktivieren implementieren 🎬 Implementation im LocationListener: alle GPS-Meldungen im Textfenster ausgeben. 🎬 Alles mit dem GPS-Emulator testen 🎬 Danach implementieren sie entsprechend der Diskussion in der Vorlesung die Konvertierung von GPS-Koordinaten in das Pixel-Koordinatensystem von FHFL_Map. Geben sie auch die Pixel-Koordinaten im Debugfenster aus. 🎬 Alles mit dem GPS-Emulator testen, zeigen sie diesen Test der Laborbetreuung Jetzt die Grafik: 🎬 Binden sie die Grafik-Datei FHFL_Map.jpg in die Ressourcen Ihres Projektes ein (s. Vorlesung) 🎬 Entwerfen Sie eine eigene View Klasse, mit einem geeigneten onDraw() Member (s. Vorlesung): o Lesen sie die in der Ressource enthaltene Grafik ein und stellen sie die im Bereich des View-Objektes dar. o Übertragen des aktuellen Punktes aus der Activity in ihre View mit anschließender Darstellung in der View. 🎬 Bei jedem gültigen Koordinaten-Update wird im LocationListener der Activity nach dem Abspeichern der aktuellen GPS-Koordinaten (s.o.) der invalidate() Member des Views aufgerufen (und damit onDraw() ausgelöst). 🎬 Alles mit dem GPS-Emulator und Ihren eigenen Testdaten testen, zeigen sie diesen Test der Laborbetreuung 🎬 Versuchen sie ein Labor-Android-Smartphone zu bekommen und übertragen sie die App auf das Smartphone und testen sie auf einem Spaziergang Ihre App. Achtung: 🎬 Alle Vorgaben aus der

Vorlesung bzgl. Namensgebung und Debugging müssen (sinnvoll) implementiert werden. 🎬 Wählen sie eine vernünftige (niedrige) Aktualisierungsrate für die GPS-Daten.

Ausgabe



Quellcode

MainActivity.java

```
package com.example.labor4;

import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import android.Manifest;
import android.content.Context;
import android.content.pm.PackageManager;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
```

```
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

    String str;
    Location locationObj;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        final TextView view_debug =
(TextView)findViewById(R.id.view_debug);
        final OnView view_map = findViewById(R.id.view_map);
        final int REQUEST_PERMISSION_ACCESS_FINE_LOCATION=1;

        //Set up location manager
        final LocationManager locationManager = (LocationManager)
this.getSystemService(Context.LOCATION_SERVICE);

        final LocationListener locationListener = new
LocationListener() {
            public void onLocationChanged(Location location) {
                locationObj = location;
                // Called when a new location is found by the
network location provider.
                double[] pixel = {0,0};
                //OnView.onDraw(location);
                view_map.invalidate();
                str = "onLocationChanged(Location: " +
location.toString();

               Log.d("dbg", str);
                view_debug.setText(pixel[1] +" ;" + pixel[0] +" ;"
+ str);
            }

            public void onStatusChanged(String provider, int
status, Bundle extras) {
                Log.d("dbg", "onStatusChanged(provider: " +
provider + " status: " + status + " extras: " + extras.toString());
                Log.d("dbg", str);
                view_debug.setText(str);
            }

            public void onProviderEnabled(String provider) {
                str = "onProviderEnabled(provider: " + provider;
                Log.d("dbg", str);
                view_debug.setText(str);
```

```
            }

            public void onProviderDisabled(String provider) {
                str = "onProviderDisabled(provider: " + provider;
                Log.d("dbg", str);
                view_debug.setText(str);
            }
        };

        Button button_start = findViewById(R.id.button_start);
        button_start.setOnClickListener(new View.OnClickListener()
{

            public void onClick(View v) {
                //GPS aktivieren
                Log.d("dbg", "Active GPS");
                if
(ContextCompat.checkSelfPermission(getApplicationContext(),
Manifest.permission.ACCESS_FINE_LOCATION)
                        != PackageManager.PERMISSION_GRANTED) {
                    // Show rationale and request permission.

ActivityCompat.requestPermissions(MainActivity.this,
                            new
String[]{Manifest.permission.ACCESS_FINE_LOCATION},

REQUEST_PERMISSION_ACCESS_FINE_LOCATION);

                    Log.d("dbg", "no permission");
                }

                if
(ContextCompat.checkSelfPermission(getApplicationContext(),
Manifest.permission.ACCESS_FINE_LOCATION)
                        == PackageManager.PERMISSION_GRANTED) {
                    // Register the listener with the Location
Manager to receive location updates

locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER
,
                            0, 0, locationListener); // via GPS
                }
            }
        });

        Button button_stop = findViewById(R.id.button_stop);
        button_stop.setOnClickListener(new View.OnClickListener()
{

            public void onClick(View v) {
```

```
                //GPS deaktivieren
                Log.d("dbg", "Deactive GPS");
                locationManager.removeUpdates(locationListener);
            }
        });

        Button button_clear = findViewById(R.id.button_clear);
        button_clear.setOnClickListener(new View.OnClickListener()
{

            public void onClick(View v) {
                //Clear output
                Log.d("dbg", "Clear output");

            }
        });
    }
}
```

OnView.java

```
package com.example.labor4;

import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Rect;
import android.location.Location;
import android.util.AttributeSet;
import android.util.Log;
import android.view.View;

import androidx.annotation.Nullable;

public class OnView extends View {
    private Paint paint;
    private Bitmap bmKarte;
    private Rect rView;
    private Rect rMap;

    private double latTop = 54.776627;
    private double lngTop = 9.448113;
    private double latBot = 54.774028;
    private double lngBot = 9.453253;

    public OnView(Context context, @Nullable AttributeSet attrs) {
        super(context, attrs);
        bmKarte = BitmapFactory.decodeResource(getResources(),
```

```
R.drawable.fhfl_map);
        rMap = new Rect(0, 0, bmKarte.getWidth(),
bmKarte.getHeight());
        paint = new Paint();
        paint.setStyle(Paint.Style.FILL);
        paint.setColor(Color.RED);
    }

    @Override
    public void onDraw(Canvas canvas) {
        super.onDraw(canvas);

        rView = new Rect(0, 0, this.getWidth(), this.getHeight());
        canvas.drawBitmap(bmKarte, rMap, rView, paint);
        MainActivity activity = (MainActivity) this.getContext();

        @Nullable
        Location location = activity.locationObj;

        if (location != null) {
            Log.d("dbg",location.toString());

           //Calculate pixel coordinates
            double y = location.getLatitude();
            double x = location.getLongitude();

            double fullY = latBot - latTop;
            double fullX = lngBot - lngTop;

            double diffX = x - lngTop;
            double diffY = y - latTop;

double pixelX = diffX / fullX * this.getWidth();
double pixelY = diffY / fullY * this.getHeight();

            canvas.drawCircle((float)pixelX, (float)pixelY, 15f,
paint);
        }
    }
}
```

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
```

```
<com.example.labor4.OnView
    android:id="@+id/view_map"
    android:layout_width="0dp"
    android:layout_height="300dp"
    android:layout_marginStart="8dp"
    android:layout_marginLeft="8dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginRight="8dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:srcCompat="@drawable/fhfl_map" />

<Button
    android:id="@+id/button_stop"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="24dp"
    android:text="Stop"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/view_map" />

<Button
    android:id="@+id/button_clear"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="26dp"
    android:text="Clear"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toEndOf="@+id/button_stop"
    app:layout_constraintTop_toBottomOf="@+id/view_map" />

<Button
    android:id="@+id/button_start"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Start"
    app:layout_constraintEnd_toStartOf="@+id/button_stop"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="@+id/button_stop" />

<TextView
    android:id="@+id/view_debug"
    android:layout_width="0dp"
    android:layout_height="0dp"
    android:layout_marginStart="8dp"
    android:layout_marginLeft="8dp"
```

```
            android:layout_marginTop="24dp"
            android:layout_marginEnd="8dp"
            android:layout_marginRight="8dp"
            android:layout_marginBottom="8dp"
            android:text=""
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toBottomOf="@+id/button_stop" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Labor 5

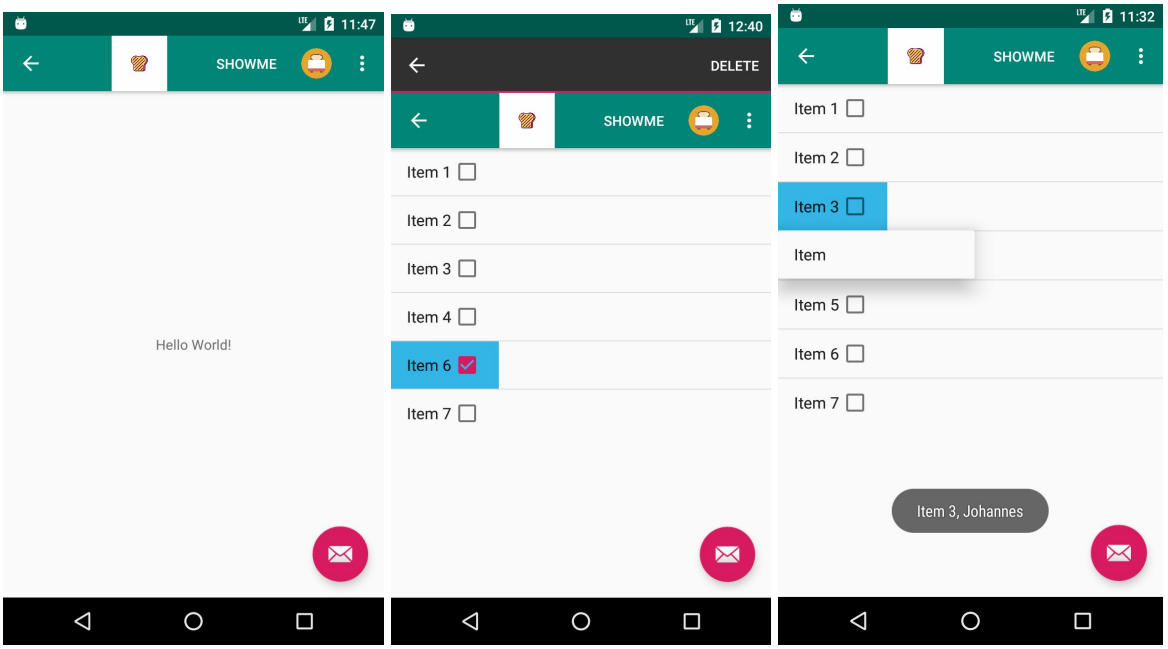
Hello Menu

Aufgabenstellung

Institut für Kommunikationstechnologie 1 Hello Menu (Sebastian Müller, 16/10-15) (Update Sebastian Müller, 01/12-15) (Update Fuchs/Schwär, 14/9-16 –v02) (Update Fuchs/Schwär, 18/9-17 – v03) (Update Fuchs/Schwär, 6/9-18 – v04) Update Fuchs/Schwär, 2/10-19- v05) Institut für Kommunikationstechnologie 2 Hello Menu In diesem Labor werden die drei Menüvarianten ActionBar, contextual action bar und PopupMenu behandelt. Ziel ist das Erlernen, wie man die unterschiedlichen Menüs einbindet, definiert und Interaktionen behandelt. Projekterstellung: Erstellen Sie ein Projekt HelloMenu (Minimum SDK API 15) mit einer Basic Activity. Institut für Kommunikationstechnologie 3 Hello Menu - Aufgabe 1a ActionBar Vorbereitung: 1. Ändern Sie in der Datei manifests/AndroidManifest.xml den Wert android:theme="@style/AppTheme" in android:theme="@style/Theme.AppCompat". Menüeinträge anlegen: 1. Öffnen Sie die XML-Menüdatei, die aktuell für die ActionBar (menu_main.xml)geladen wird. 2. Legen Sie weitere Menü-Items an: 1. showMe - soll immer angezeigt werden (‚always'). 2. showMeLong - soll im 3-Punkte-Menu oben rechts angezeigt werden (‚ifRoom'). Menü Interaktionen behandeln: Wird ein Menü-Item gedrückt, wird die onOptionsItemsSelected Methode in der MainActivity aufgerufen. Nach erfolgreich behandelter Interaktion gibt diese Methode true zurück. 1. Beim Drücken eines Menü-Items soll ein kurzer Toast ausgegeben werden, in dem der Text des jeweiligen Items steht. 2. Beim Drücken des showMeLong-Items soll ein langer Toast mit derselben Nachricht ausgegeben werden. Institut für Kommunikationstechnologie 4 Hello Menu - Aufgabe 1a ActionBar Bild für ein Menü-Item festlegen: 1. Legen Sie ein weiteres Menü-Item an, das immer angezeigt wird. 2. Weisen sie diesem Item ein icon über das Attribut android:icon="@drawable/..." zu. 3. Beim Drücken auf dieses Icon soll ebenfalls ein kurzer Toast mit dem Text des Items ausgegeben werden. Starten sie die Applikation Institut für Kommunikationstechnologie 5 Hello Menu - Aufgabe 1b ActionBar Logo der ActionBar festlegen: Damit ein Logo, statt des Namens der Applikation in der ActionBar angezeigt wird, gehen Sie wie folgt vor: 1. In der onCreateOptionsMenu Methode der MainActivity holen Sie sich zuerst die Actionbar (getSupportActionBar();) 2. Blenden Sie für die ActionBar den Titel aus. (actionBar.setDisplayShowTitleEnabled(false);) Um das gewünschte Logo einzublenden gehen Sie wie folgt vor: 1. actionBar.setIcon(R.drawable.....); 2. actionBar.setDisplayUseLogoEnabled(true); Um zusätzlich einen „zurück-Pfeil" zu erzeugen, gehen Sie wie folgt vor: 3. actionBar.setDisplayHomeAsUpEnabled(true); Starten sie die Applikation Institut für Kommunikationstechnologie 6 Hello Menu - Aufgabe 2 contextual action bar Für diese Aufgabe wird ein ListView verwendet. Langes Drücken auf ein ListItem hat das Öffnen der contextual action bar (cab) zur Folge. Es können mehrere Listenelemente markiert werden, um so dieselbe Aktion für alle markierten Listenelemente auszuführen. Vorbereitung: 1. Binden Sie die zur Verfügung gestellte Datei ListViewHelper.java in Ihr Projekt ein. (Passen Sie, falls nötig, das Package in der ListViewHelper.java Datei an ihr Package an) 2. Legen sie in der content_main.xml ein ListView Element an. 3. Holen Sie sich in der MainActivity diesen ListView. 4. Rufen Sie ListViewHelper.initAdapter(Context, ListView) auf. 5. Implementieren Sie in der MainActivity einen MutliChoiceModeListener und setzen Sie diesen für den ListView. Contextual action bar definieren: 1. Legen Sie eine neue Menüdatei in dem Verzeichnis res/menu an. 2. Legen Sie ein Menü-Item „Delete" an. Institut für Kommunikationstechnologie 7 Hello Menu - Aufgabe 2 contextual action bar (cab) Interaktionen mit der contextual action bar behandeln: 1. Nutzerinteraktionen mit der cab werden innerhalb des MultiChoiceModeListener behandelt. 2. Bei langem Drücken auf ein Listenelement wird die Methode

onCreateActionMode aufgerufen. 3. Erzeugen sie diese Methode und laden sie in dieser Methode mit Hilfe des MenuInflaters das erstellte Menü. 4. Wird ein Menü-Item lange gedrückt, wird die Methode onActionItemClicked aufgerufen. Behandeln Sie dort die Interaktionen mit dem „Delete“-Item. Rufen Sie dabei ListViewHelper.removeCheckedItems() auf. 5. Anschließend rufen Sie die entsprechende Methode des übergebenen ActionMode auf, die kennzeichnet, dass der contextual action mode fertig ist. (Der contextual action mode wird deaktiviert und die ursprüngliche ActionBar ist wieder zu sehen) 6. Beachten sie, dass die Methode onItemClick vorhanden sein muss. Starten Sie die App. Das Bild müsste dann so aussehen. Langes Drücken auf ein Item führt zum Erscheinen des Feldes „CAB_DELETE“ und zur Markierung des Feldes. Institut für Kommunikationstechnologie 8 Hello Menu - Aufgabe 3 PopupMenu Es wird ein PopupMenu für jedes Listenelement definiert. Beim einfachen Druck auf ein Listenelement öffnet sich dieses Menü. Vorbereitung: 1. Implementieren sie in der MainActivity einen onItemClickListener und setzen Sie diesen für Ihren listView. 2. Die zugehörige onClick-Methode wird aufgerufen, sobald man kurz auf ein Listenelement klickt. Definition des Popup Menüs: 1. Legen Sie eine neue Menüdatei in dem Verzeichnis res/menu an. 2. Legen Sie ein Menü-Item an, dass beim kurzen Drücken auf ein Item den Aufruf des Popup-Menus kennzeichnet. Institut für Kommunikationstechnologie 9 Hello Menu - Aufgabe 3 PopupMenu Popup Menü anzeigen: Beim Drücken eines Listenelementes soll ein neues PopupMenu erstellt und angezeigt werden. 1. Erstellen sie beim Klick auf ein Listenelement (onItemClick) ein neues PopupMenu. (new PopupMenu(MainActivity.this, view)); 2. Nutzen Sie den MenuInflater des Popup-Objektes, um mit der Methode inflate() das PopupMenu zu laden. (popup.getMenuInflater().inflate(R.menu.MenuLayoutDatei Name, popupMenu.getMenu()) Nutzerinteraktionen behandeln: 1. Anschließend muss nun, wenn ein MenuItem geklickt wurde, die entsprechende Methode aufgerufen wird, um das PopupMenu anzuzeigen. 2. Implementieren Sie das OnMenuItemClickListener Interface und setzen Sie dieses für das PopupMenu. 3. Die darin enthaltene onMenuItemClick Methode wird aufgerufen, sobald ein Menüelement gedrückt wird. 4. Geben Sie einen kurzen Toast aus, der den Namen des Menüelementes und Ihren eigenen ausgibt.

Ausgabe



Quellcode

MainActivity.java

```
package com.example.hellomenu;

import android.app.Activity;
import android.os.Bundle;

import
com.google.android.material.floatingactionbutton.FloatingActionBut
ton;
import com.google.android.material.snackbar.Snackbar;

import androidx.appcompat.app.ActionBar;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;

import android.util.Log;
import android.util.SparseBooleanArray;
import android.view.ActionMode;
import android.view.MenuInflater;
import android.view.View;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.AbsListView;
import android.widget.AdapterView;
import android.widget.ListView;
import android.widget.PopupMenu;
import android.widget.Toast;
import com.example.hellomenu.ListViewHelper;

import static com.example.hellomenu.ListViewHelper.adapter;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Toolbar toolbar = findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        FloatingActionButton fab = findViewById(R.id.fab);
        fab.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Snackbar.make(view, "Replace with your own action",
Snackbar.LENGTH_LONG)
                        .setAction("Action", null).show();
            }
```

```
        });

        final ListView listView = (ListView)
findViewById(R.id.listview);
        ListViewHelper.initAdapter(this, listView);

        listView.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, View
view, int position, long id) {
                Log.d("dbg","onItemClick");

                PopupMenu popup = new PopupMenu(MainActivity.this,
view);
                MenuInflater inflater = popup.getMenuInflater();
                inflater.inflate(R.menu.menu_popup,
popup.getMenu());
                popup.show();
                String name = (String)
parent.getItemAtPosition(position);

                Toast.makeText(getApplicationContext(),name + ",
Johannes", Toast.LENGTH_SHORT).show();
            }
            /*
            //@Override
            public boolean onMenuItemClick(MenuItem item) {
                Log.d("dbg","onPopupMenuClick");
                return true;
            }
            */
        });

        listView.setMultiChoiceModeListener(new
AbsListView.MultiChoiceModeListener() {
            @Override
            public boolean onCreateActionMode(ActionMode mode, Menu
menu) {
                Log.d("dbg","onCreateActionMode");
                MenuInflater inflater = getMenuInflater();
                inflater.inflate(R.menu.menu_delete, menu);
                return true;
            }

            @Override
            public boolean onPrepareActionMode(ActionMode mode,
Menu menu) {
                return false;
            }
```

```
            @Override
            public boolean onActionItemClicked(ActionMode mode,
MenuItem item) {
                ListViewHelper.removeCheckedItems(listView);
                final ActionMode actionMode = mode;
                actionMode.finish();
                return true;
            }

            @Override
            public void onDestroyActionMode(ActionMode mode) {

            }

            @Override
            public void onItemCheckedStateChanged(ActionMode mode,
int position, long id, boolean checked) {
            }
        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it
is present.
        getMenuInflater().inflate(R.menu.menu_main, menu);
        ActionBar actionBar = getSupportActionBar();
        actionBar.setDisplayShowTitleEnabled(false);
        actionBar.setIcon(R.drawable.toast2);
        actionBar.setDisplayUseLogoEnabled(true);
        actionBar.setDisplayHomeAsUpEnabled(true);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so
long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
        Log.d("dbg", (String.valueOf(id)));

        //noinspection SimplifiableIfStatement
        switch (id) {
            case R.id.action_settings:
                return true;
            case R.id.showMe:
                Toast.makeText(getApplicationContext(),"ShowMe",
```

```
Toast.LENGTH_SHORT).show();
                return true;
            case R.id.image:
                Toast.makeText(getApplicationContext(),"Image",
Toast.LENGTH_SHORT).show();
                return true;
            case R.id.showMeLong:

Toast.makeText(getApplicationContext(),"ShowMeLong",
Toast.LENGTH_LONG).show();
                return true;
            default:
                Log.d("dbg","Unknown Menu");
        }

        return super.onOptionsItemSelected(item);
    }
}
```

ListViewHelper.java

```
package com.example.hellomenu;

import android.content.Context;
import android.util.Log;
import android.util.SparseBooleanArray;
import android.widget.AbsListView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collection;

/**
 * Helper class for the HelloMenu assignment.
 * Created by Sebastian Mueller on 25.10.15.
 */
public class ListViewHelper {

    /** Logcat TAG*/
    private static final String TAG = "ListViewHelper";
    /** Simple ArrayAdapter<String>*/
    public static ArrayAdapter<String> adapter;
    /** ArrayList of Strings*/
    private static ArrayList<String> values = new
ArrayList<>(Arrays.asList(
            "Item 1", "Item 2", "Item 3", "Item 4", "Item 5", "Item
6", "Item 7"));

    /**
     * Initializes the ArrayAdapter and sets the adapter for the
```

```
given list view.
     *
     * @param context - context
     * @param listView - list view for which the adapter should be
set
     */
    public static void initAdapter(Context context, ListView
listView) {
        Log.i(TAG, "initAdapter()");
        //Create Adapter
        adapter = new ArrayAdapter<String>(context,
                android.R.layout.simple_list_item_multiple_choice,
android.R.id.text1, values);

        // Assign adapter to ListView
        listView.setAdapter(adapter);

listView.setChoiceMode(AbsListView.CHOICE_MODE_MULTIPLE_MODAL);
        listView.setSelector(android.R.color.holo_blue_light);
    }

    /**
     * Removes all checked items from the given ListView
     * @param listView - listView
     */
    public static void removeCheckedItems(ListView listView) {
        Log.i(TAG, "removeCheckedItems");
        SparseBooleanArray checkedItems =
listView.getCheckedItemPositions();
        Collection<Object> removeAbleItems = new
ArrayList<Object>();

        //Iteratation over all checked items (checked and unchecked
are in checkedItems)
        String str = "";
        for (int i = 0; i < checkedItems.size(); i++) {
            if (checkedItems.valueAt(i)) {
                str += checkedItems.keyAt(i) + ", ";

removeAbleItems.add(listView.getItemAtPosition(checkedItems.keyAt(
i)));
            }
        }

        //Remove all checked items from list
        values.removeAll(removeAbleItems);
        //Notify
        adapter.notifyDataSetChanged();
        Log.d(TAG, "Deleted items at Position: " + str);
    }
```

```
}
```

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.coordinatorlayout.widget.CoordinatorLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <com.google.android.material.appbar.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/AppTheme.AppBarOverlay">

        <androidx.appcompat.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="?attr/colorPrimary"
            app:popupTheme="@style/AppTheme.PopupOverlay" />

    </com.google.android.material.appbar.AppBarLayout>

    <include layout="@layout/content_main" />


<com.google.android.material.floatingactionbutton.FloatingActionButt
on
        android:id="@+id/fab"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="bottom|end"
        android:layout_margin="@dimen/fab_margin"
        app:srcCompat="@android:drawable/ic_dialog_email" />

</androidx.coordinatorlayout.widget.CoordinatorLayout>
```

ContentMain.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context=".MainActivity"
```

```
    tools:showIn="@layout/activity_main">

    <ListView
        android:id="@+id/listview"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:layout_constraintEnd_toEndOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```