Tutorial

Introduction to Android App Development

Software Engineering
Prof. K. M. Berkling, Ph.D.
TINF17B2

e-Portfolio
Nils Krehl

24. October 2018

Table of contents

1. Introduction

In this tutorial you are going to built your first Android App and you are going to learn the key concepts of defining Layouts and Application Programming for Android Apps. The example project prepared for this tutorial is choosen carefully to get acquainted to central concepts in a short timeframe. You are going to design your first UI, handle user input and trigger a screen change in your App.

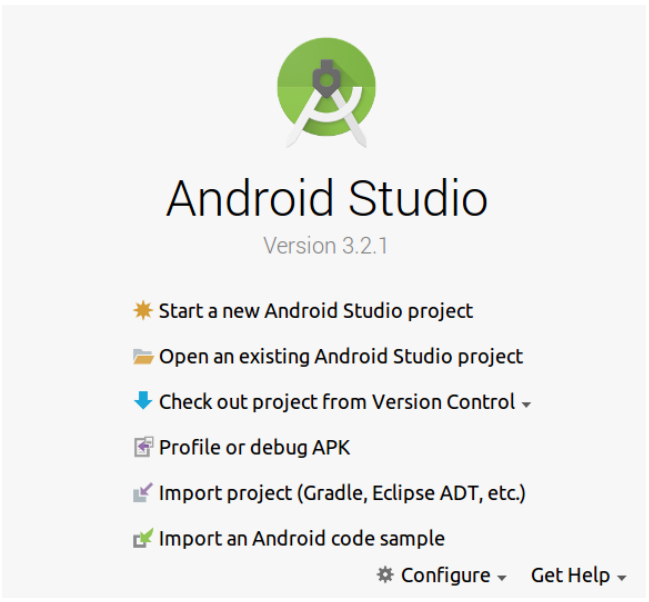
2. Setup

For developing Android Apps you need Google´s official IDE Android Studio. You can find it under the following link:
https://developer.android.com/studio/install

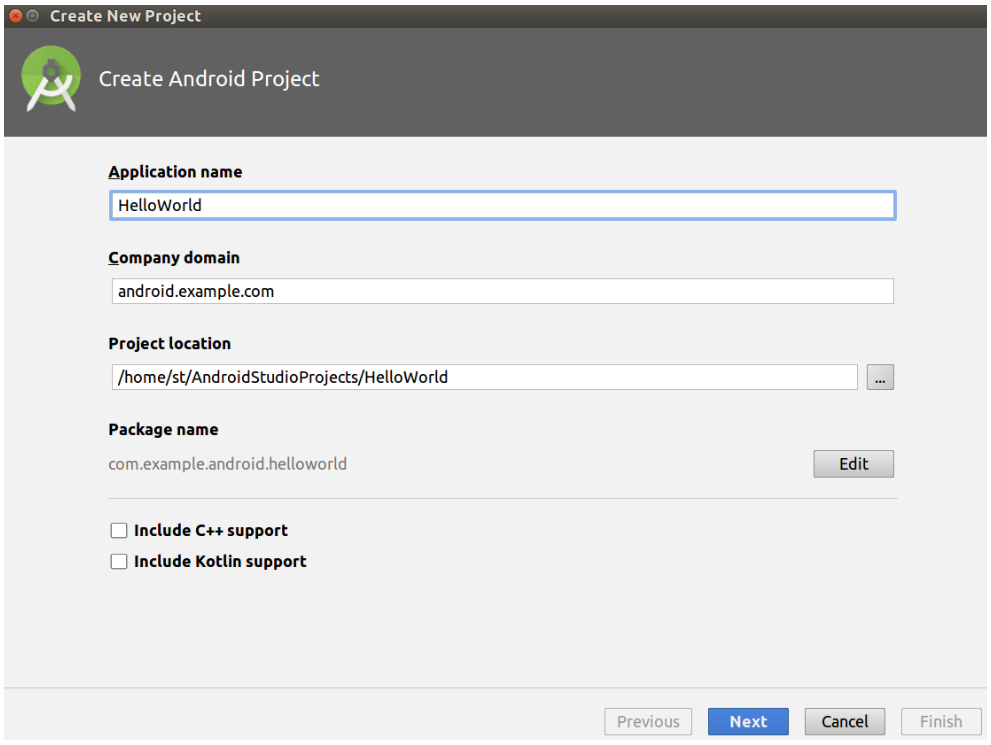
Inside of Android Studio please make sure, that you have set up an Emulator for testing your Apps. Further information on setting up an emulator correctly are accessible here: https://developer.android.com/studio/run/managing-avds

3. Start a Project

1. Select "Start an new Android Studio project"


Android Studio
Version 3.2.1
Start a new Android Studio project
Open an existing Android Studio project
Check out project from Version Control
Profile or debug APK
Import project (Gradle, Eclipse ADT, etc.)
Import an Android code sample
Configure
Get Help

2. Select Application Name



Application name

HelloWorld

Company domain

android.example.com

Project location

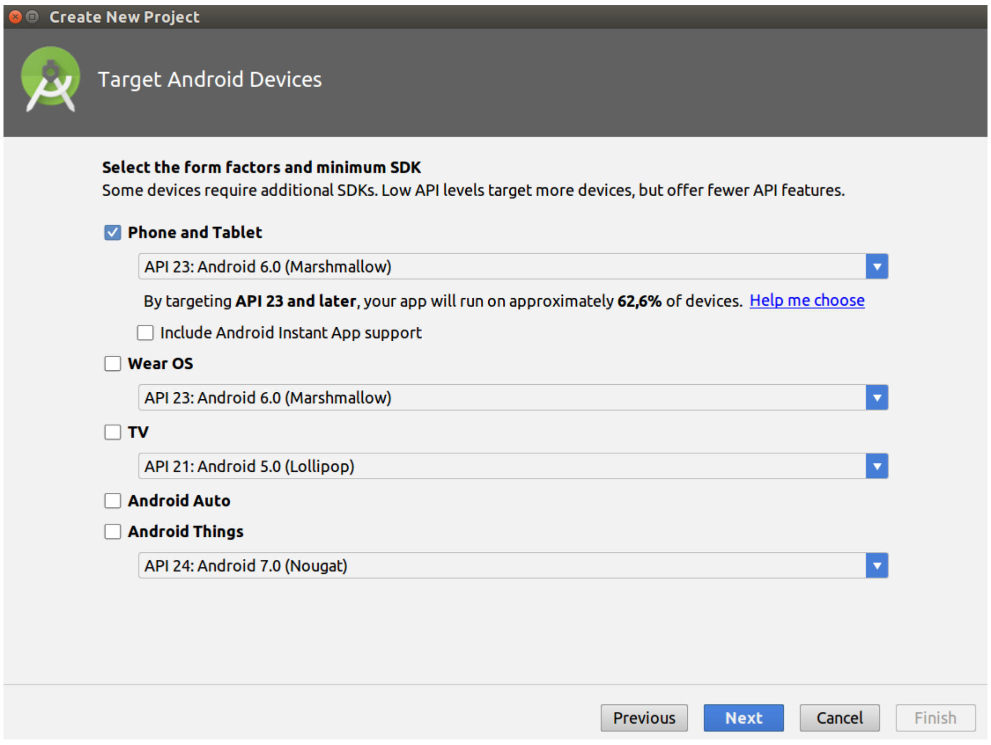
/home/st/AndroidStudioProjects/HelloWorld

Package name

com.example.android.helloworld

- [ ] Include C++ support
- [ ] Include Kotlin support

3. Select target Android Devices

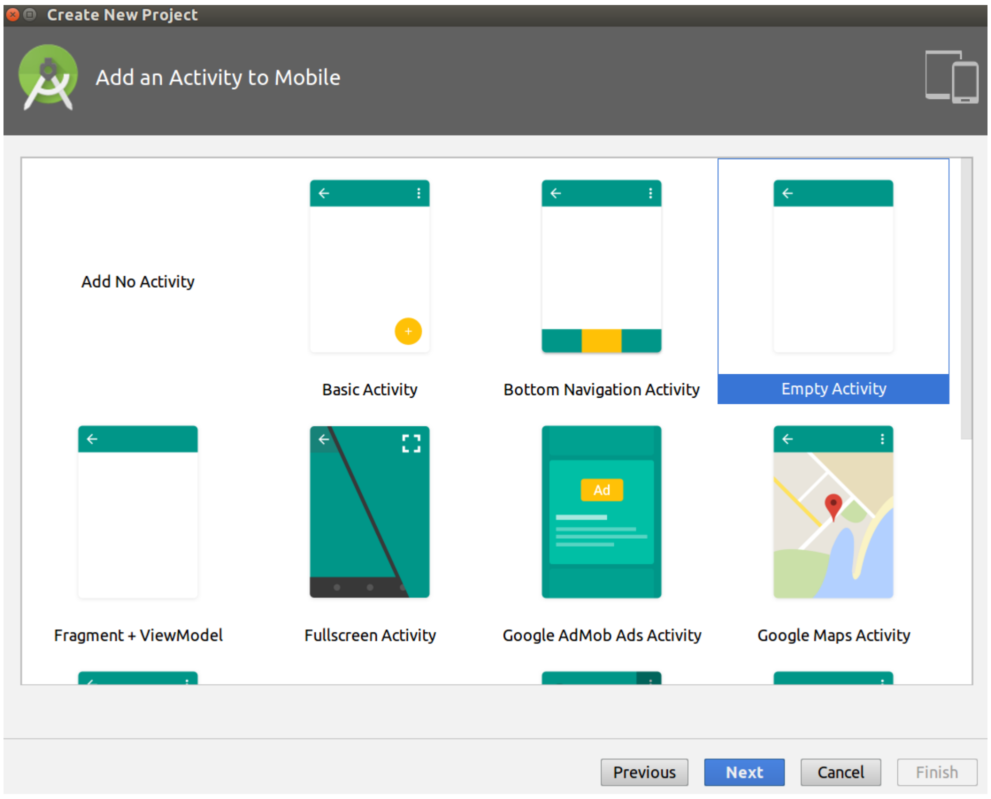


Select the form factors and minimum SDK

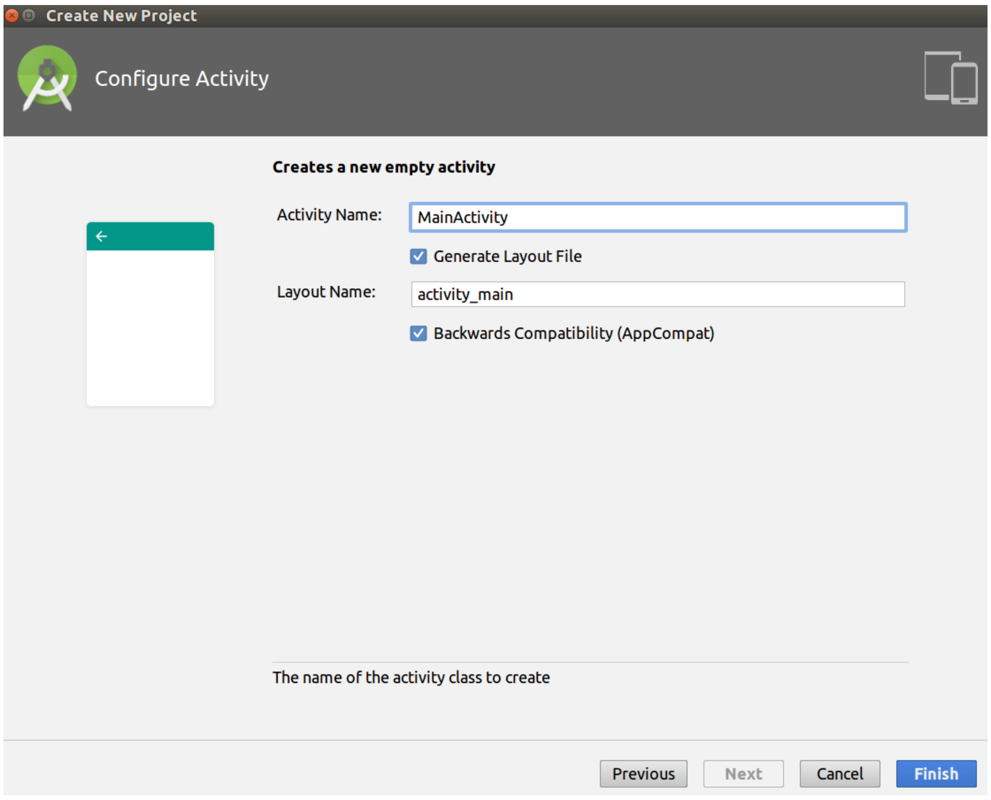
Some devices require additional SDKs. Low API levels target more devices, but offer fewer API features.

- [x] Phone and Tablet
  - API 23: Android 6.0 (Marshmallow)
  - By targeting **API 23 and later**, your app will run on approximately **62,6%** of devices. Help me choose
  - [ ] Include Android Instant App support
- [ ] Wear OS
  - API 23: Android 6.0 (Marshmallow)
- [ ] TV
  - API 21: Android 5.0 (Lollipop)
- [ ] Android Auto
- [ ] Android Things
  - API 24: Android 7.0 (Nougat)

4. Add an Activity



5. Configure Activity



Creates a new empty activity

Activity Name: MainActivity

☑ Generate Layout File

Layout Name: activity_main

☑ Backwards Compatibility (AppCompat)

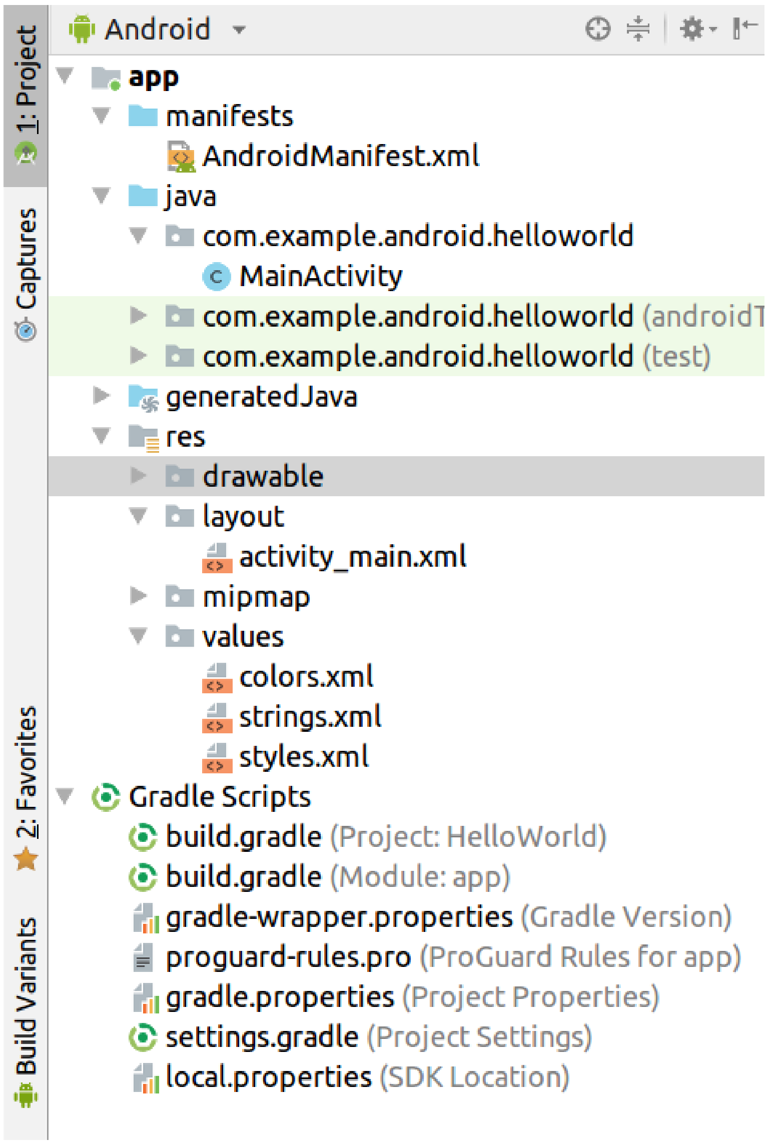
The name of the activity class to create

4. Project Structure

The typical files of a Android project can be seen in the screenshot. A description about the different files can be found here:
https://developer.android.com/studio/projects/

An overview about the `/res` directory can be found here:
https://developer.android.com/guide/topics/resources/providing-resources

In this tutorial we are going to work inside of the `/java` and `/res/layout` directories. Furthermore we have a look at the `/manifests/AndroidManifest.xml`.

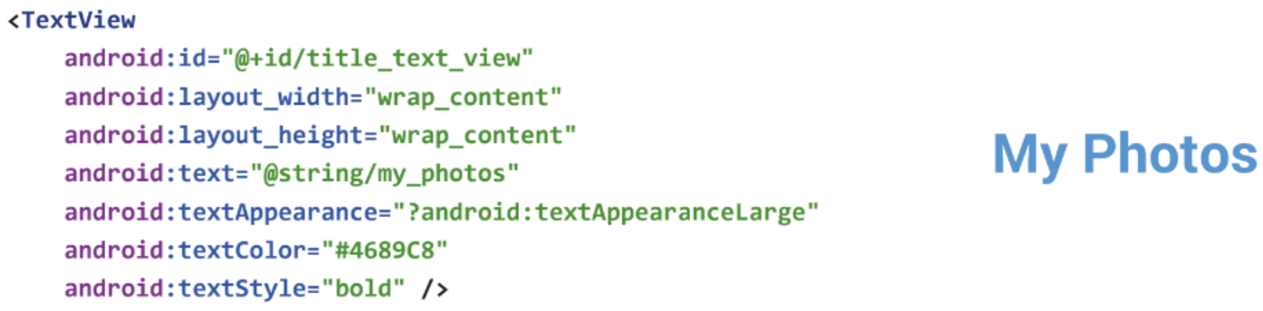


```
▼ app
    ▼ manifests
        AndroidManifest.xml
    ▼ java
        ▼ com.example.android.helloworld
            MainActivity
        ▶ com.example.android.helloworld (androidT
        ▶ com.example.android.helloworld (test)
    ▶ generatedJava
    ▼ res
        ▶ drawable
        ▼ layout
            activity_main.xml
        ▶ mipmap
        ▼ values
            colors.xml
            strings.xml
            styles.xml
▼ Gradle Scripts
    build.gradle (Project: HelloWorld)
    build.gradle (Module: app)
    gradle-wrapper.properties (Gradle Version)
    proguard-rules.pro (ProGuard Rules for app)
    gradle.properties (Project Properties)
    settings.gradle (Project Settings)
    local.properties (SDK Location)
```

5. User Interface Definition

In Android the User Interface is described with XML Files.
An introduction into the process of building a User Interface can be found here:
https://developer.android.com/guide/topics/ui/declaring-layout
Under this link you can find a Cheat Sheet with common Android Views:
https://drive.google.com/file/d/0B5XIkMkayHgRMVljUVIyZzNmQUU/view

The following code snippet shows an example of a TextView defined in XML.



```
<TextView
    android:id="@+id/title_text_view"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/my_photos"
    android:textAppearance="?android:textAppearanceLarge"
    android:textColor="#4689C8"
    android:textStyle="bold" />
```

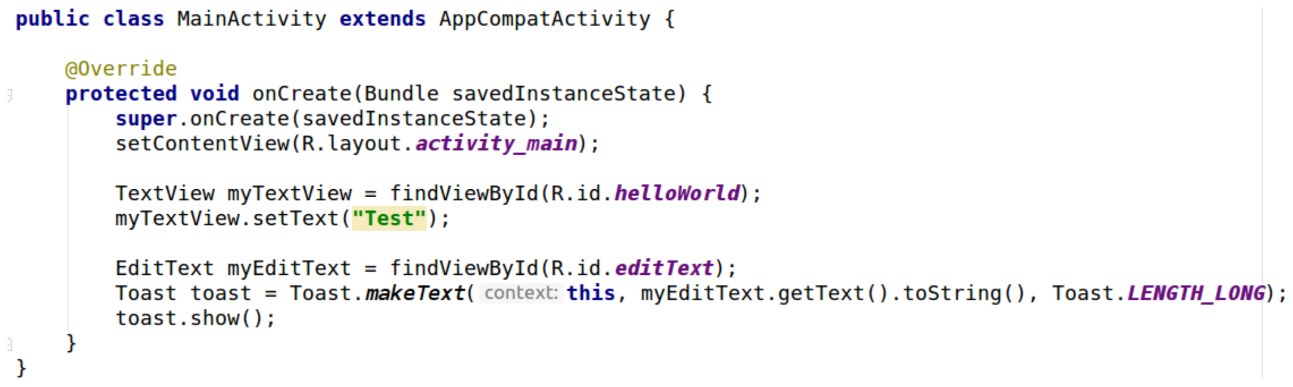
You can write these files manually or use the Android Studio UI Builder and drag and drop the UI elements onto the screen.

6. Application Programming

An introduction into programming Android Apps can be found here: https://developer.android.com/training/basics/firstapp/. In this tutorial you are going to build your first App and learn the basics of Android App development.

An Android App project consists of Java classes for the Activities (Logic for UI, i.e. Controller for the View in the XML files) and other more problem domain specific Java classes (Model).

The following code snippet shows the `onCreate` method (one of the lifecycle methods) of one Activity. This method is called from the OS when the Activity is launched. In this method the correspondent layout file is referenced (`setContentView`).



```java
public class MainActivity extends AppCompatActivity {

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

        TextView myTextView = findViewById(R.id.helloWorld);
        myTextView.setText("Test");

        EditText myEditText = findViewById(R.id.editText);
        Toast toast = Toast.makeText( context: this, myEditText.getText().toString(), Toast.LENGTH_LONG);
        toast.show();
    }
}
```

Based on the first tutorial there is a more complex introduction into the development of an Android App with two Activities and a Button OnClickListener:
https://developer.android.com/training/basics/firstapp/starting-activity

7. Hands on

The code for this tutorial can be found here: https://github.com/nilskre/ExampleApp.

The different branches contain the result code for each step:

7.1. → https://github.com/nilskre/ExampleApp/tree/1_helloWorld

7.2. → https://github.com/nilskre/ExampleApp/tree/2_UI_Elements

7.3. → https://github.com/nilskre/ExampleApp/tree/3_interaction

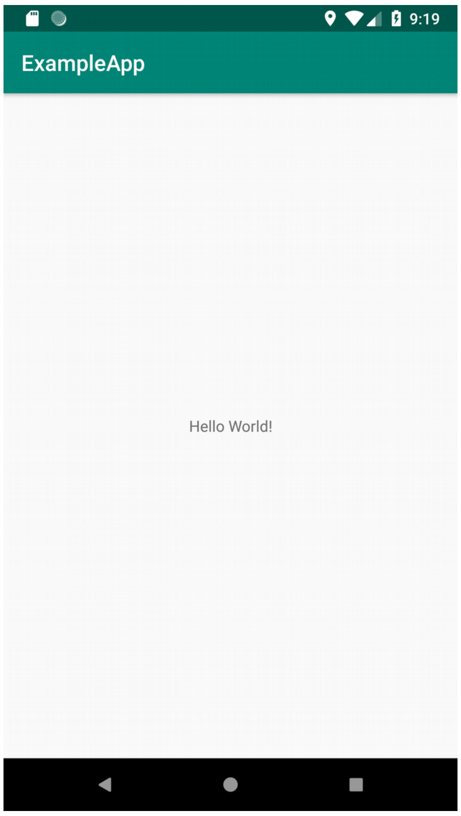
7.4. → https://github.com/nilskre/ExampleApp/tree/4_newActivity

7.5. → https://github.com/nilskre/ExampleApp/tree/5_showSecondActivity

7.1. New Project

Task: Create a new project with one TextView and run it on the Emulator.

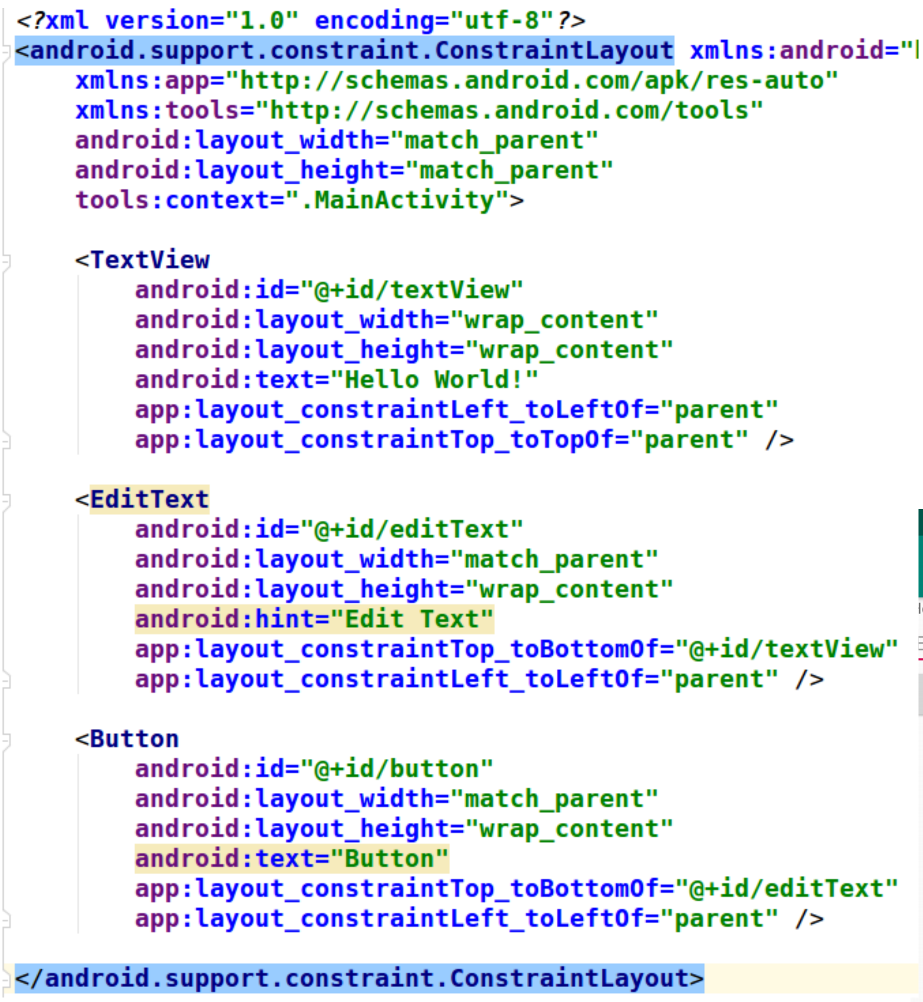
For creating a new project follow the steps in chapter three.
After creating a new project the layout file `activity_main.xml` and the `MainActivity.java` file are generated. When running the App without any code changes on a device or Emulator it looks like this:



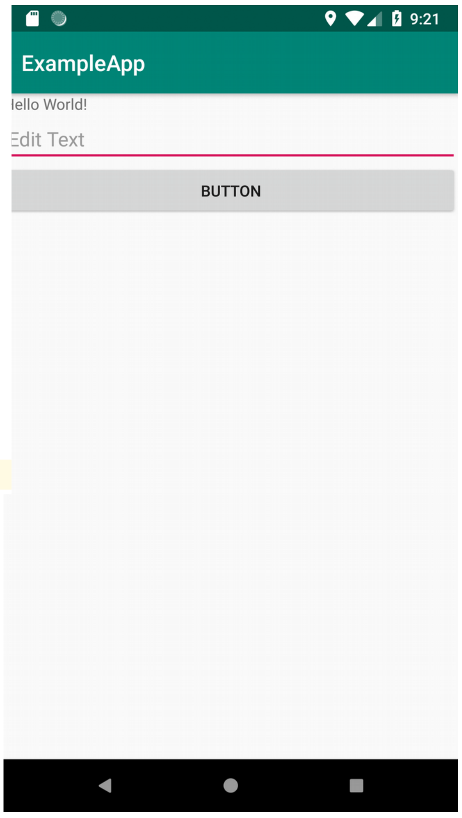
7.2. Define UI elements on screen

Task: Add new UI Elements (in the `activity_main.xml` file) to the screen:
- TextView
- EditText
- Button

In the following code snippet you can see the file `activity_main.xml` with all necessary changes. And a screenshot with the resulting App.



```xml
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns: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/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/editText"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Edit Text"
        app:layout_constraintTop_toBottomOf="@+id/textView"
        app:layout_constraintLeft_toLeftOf="parent" />

    <Button
        android:id="@+id/button"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Button"
        app:layout_constraintTop_toBottomOf="@+id/editText"
        app:layout_constraintLeft_toLeftOf="parent" />

</android.support.constraint.ConstraintLayout>
```



7.3. Interaction

Task: Set TextView with the input of EditText:
- add Button onclickListener
- String of EditText input
- set Text of TextView to String

The necessary code changes are all inside of the `MainActivity.java` file.

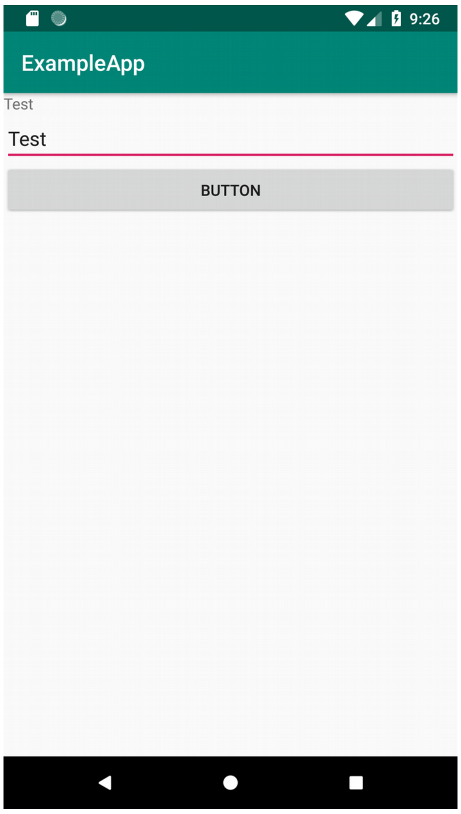
The code snippet stores the TextView inside a variable, so we can access it inside of the `MainActivity.java`.

```
TextView myTextView = (TextView)findViewById(R.id.textView);
```

The following code snippet shows a button onclick listener and sets the text of the TextView:

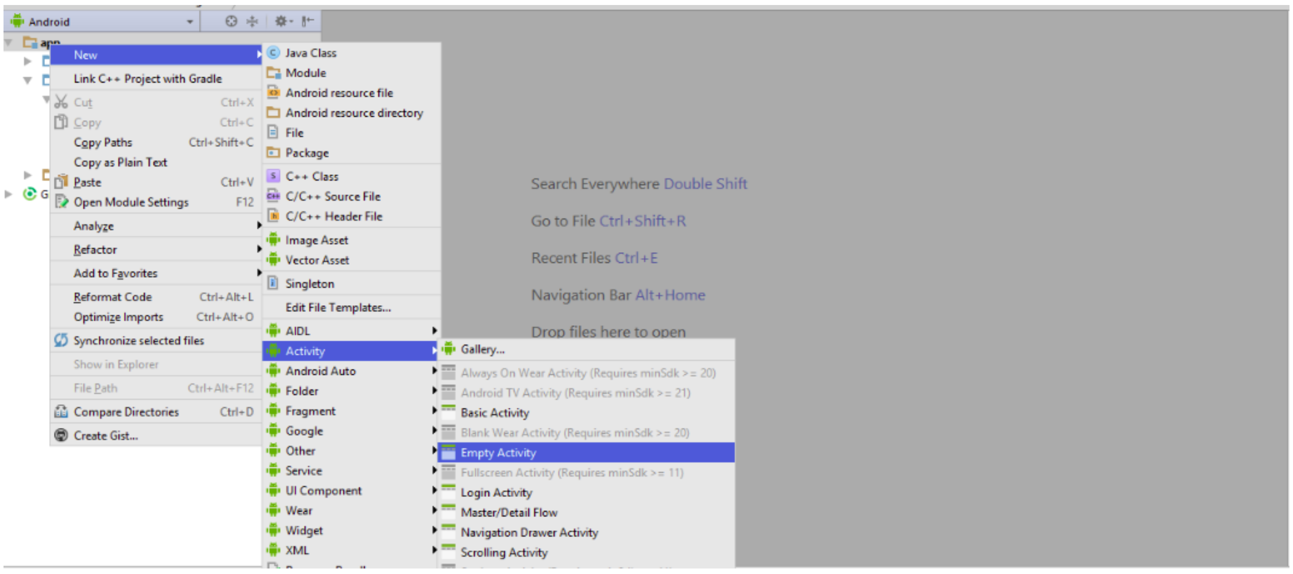
```
Button myButton = (Button)findViewById(R.id.button);
myButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        EditText myEditText = (EditText)findViewById(R.id.editText);
        String input = myEditText.getText().toString();
        myTextView.setText(input);
    }
});
```

The result should look like this:



7.4. Second Activity

Task: Add second Activity.

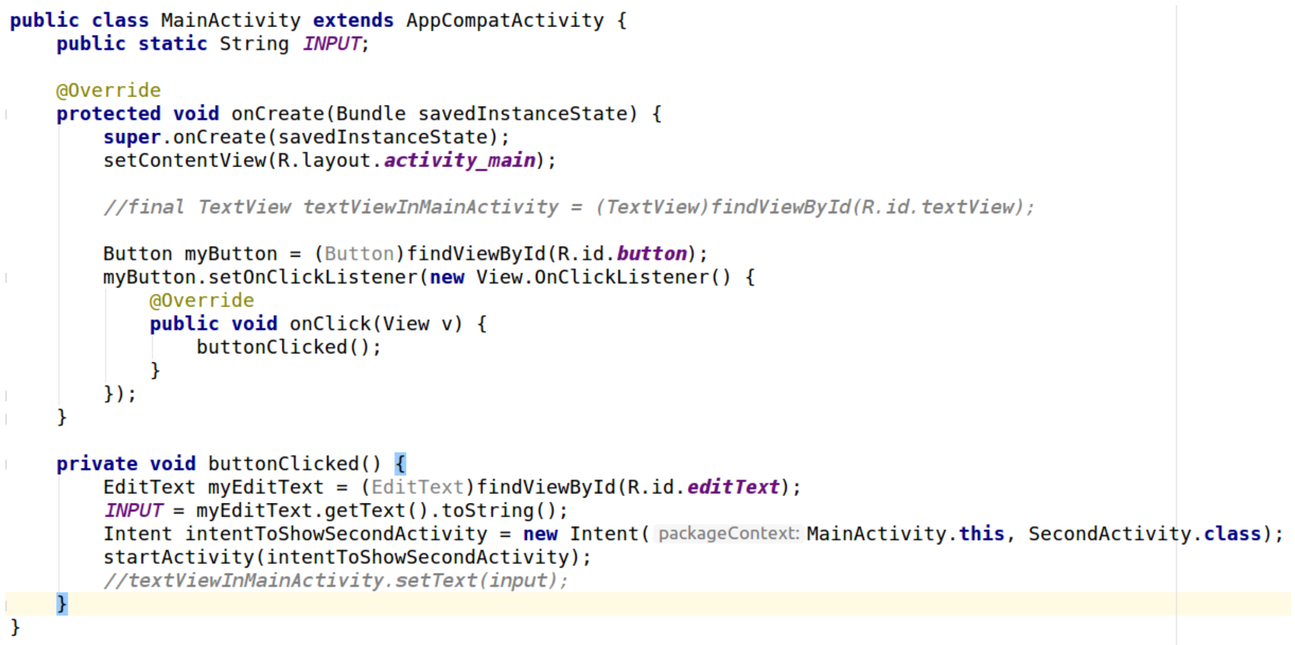


A new Activity gets created with Layout file and Java Activity file.
The new activity isn´t visible at the moment.

7.5. Switch to second Activity

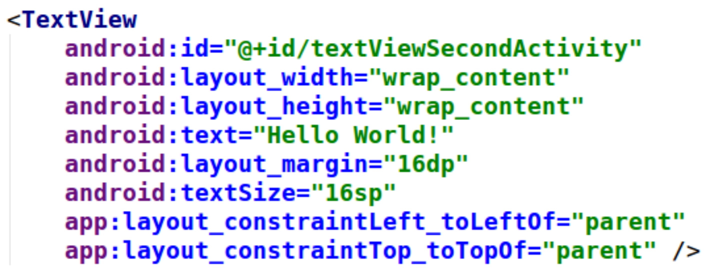
Task: Show second activity and set TextView to EditText input:
- show second Activity on Button click (Intent)
- adapt TextView in second Activity
- set TextView in second Activity

The `MainActivity` shows the second Activity after a Button click and stores the input of the EditText inside of a variable (to reach it from another Activity).



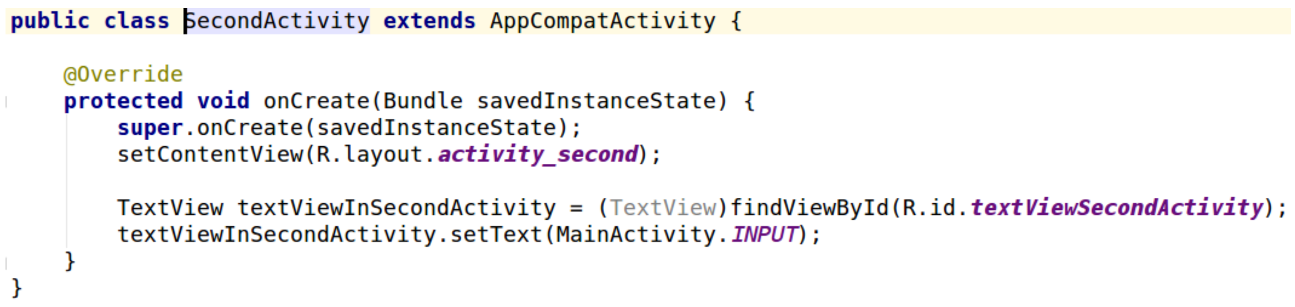
```java
public class MainActivity extends AppCompatActivity {
    public static String INPUT;

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

        //final TextView textViewInMainActivity = (TextView)findViewById(R.id.textView);

        Button myButton = (Button)findViewById(R.id.button);
        myButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                buttonClicked();
            }
        });
    }

    private void buttonClicked() {
        EditText myEditText = (EditText)findViewById(R.id.editText);
        INPUT = myEditText.getText().toString();
        Intent intentToShowSecondActivity = new Intent( packageContext: MainActivity.this, SecondActivity.class);
        startActivity(intentToShowSecondActivity);
        //textViewInMainActivity.setText(input);
    }
}
```

In the layout file of the second Activity the TextView needs an ID (textViewSecondActivity) to acccess it inside of the Java code.



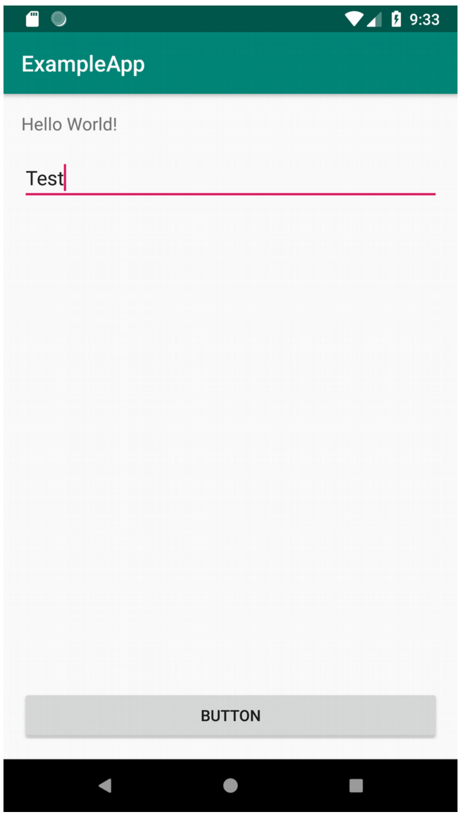
```xml
<TextView
    android:id="@+id/textViewSecondActivity"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Hello World!"
    android:layout_margin="16dp"
    android:textSize="16sp"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

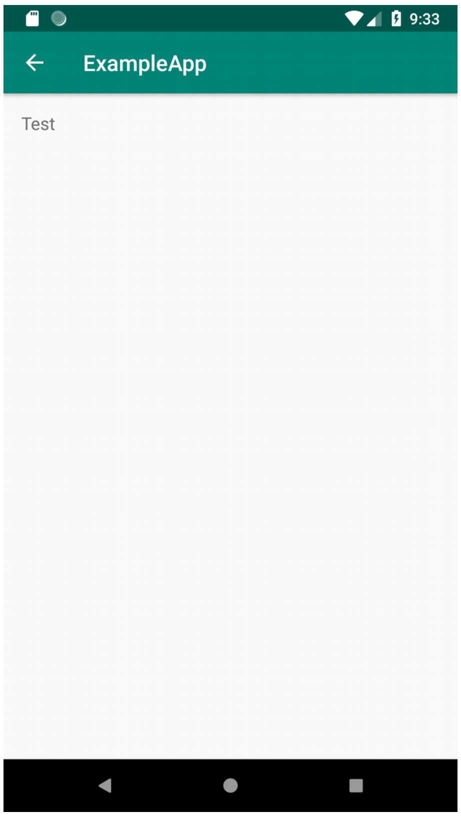
The TextView of the second Activity is set by an Intent in the `onCreate` method.



```java
public class SecondActivity extends AppCompatActivity {

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

        TextView textViewInSecondActivity = (TextView)findViewById(R.id.textViewSecondActivity);
        textViewInSecondActivity.setText(MainActivity.INPUT);
    }
}
```

The result looks like this:





8. Conclusion

Congratulations, you have written your first Android App! While doing that you have discovered the basic concepts of Android App development.

If you want to dive deeper into this topic you can follow the link to a Udacity course for advanced Android development. There you are going to learn how to fetch data from the web, how to persist data on a device and many more: https://eu.udacity.com/course/new-android-fundamentals--ud851