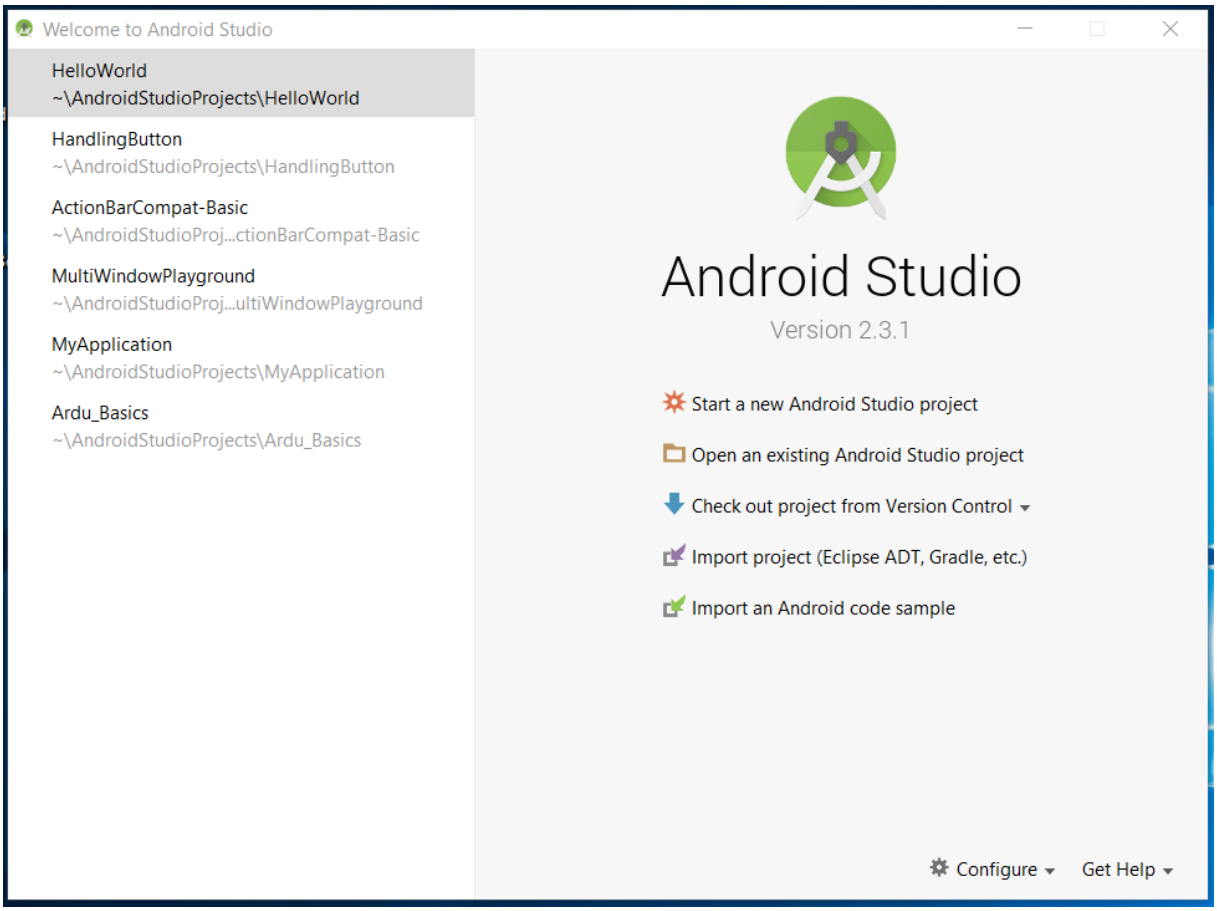
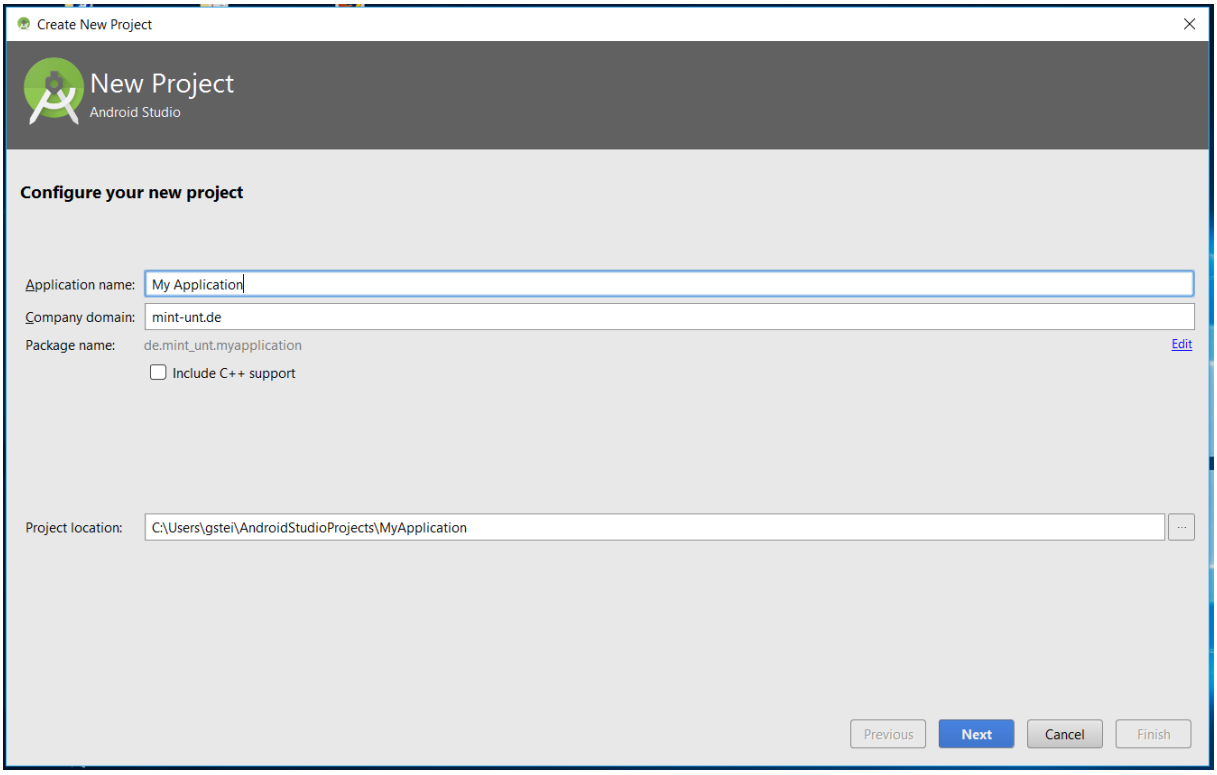
Mit Android Studio Seiten mit Text, Bildern Schaltknöpfen generieren, um zu den Seiten und zum Internet navigieren zu können.

Stufe 1:

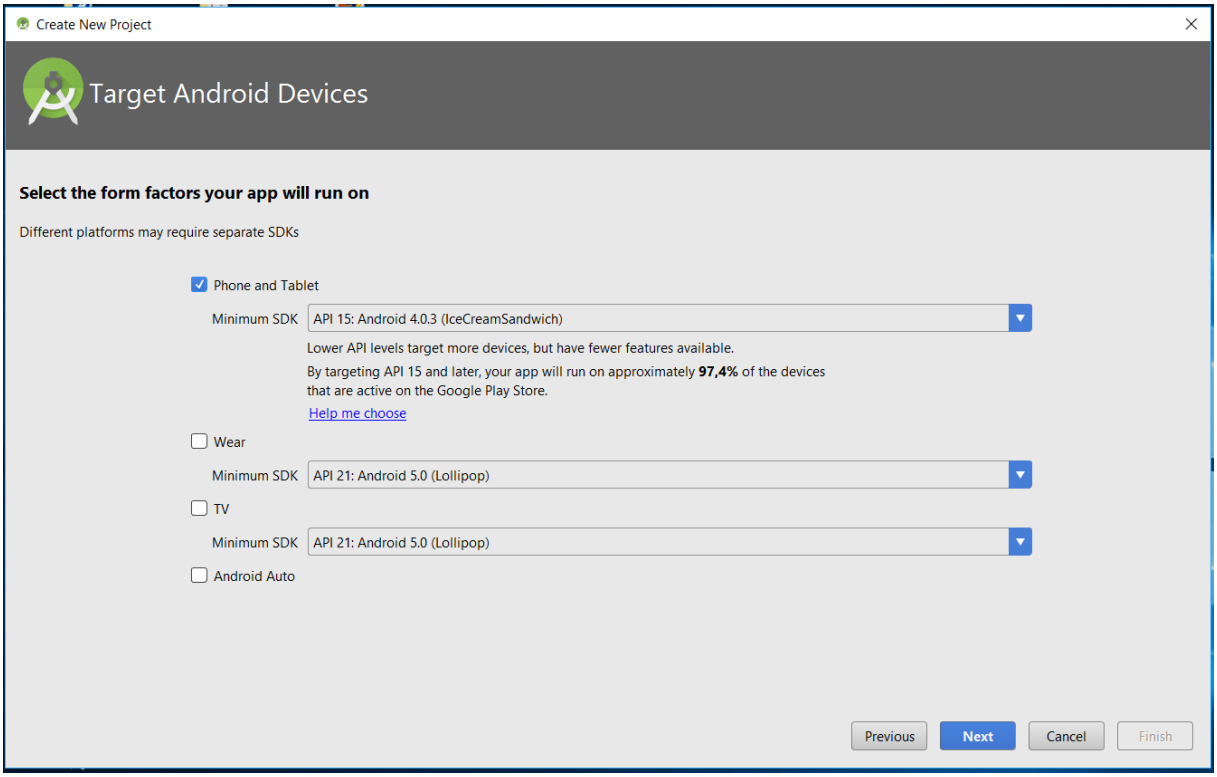
Gehe zu http://developer.android.com/sdk/index.html, lade „Android Studio“ herunter, installiere das Programm und starte ein neues Projekt.



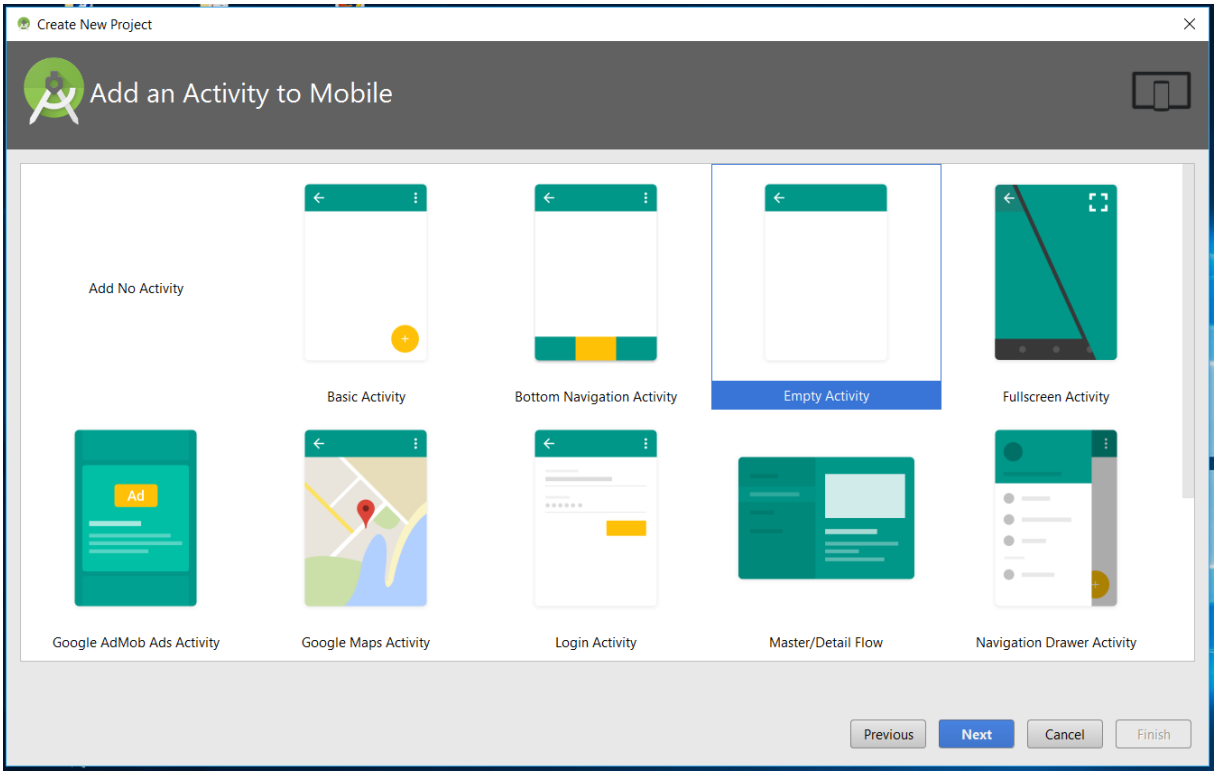
Gib dem Projekt (z.B.) den Namen „MyApplication“ und klicke auf „Next“. Der Domain-Name ist beliebig.



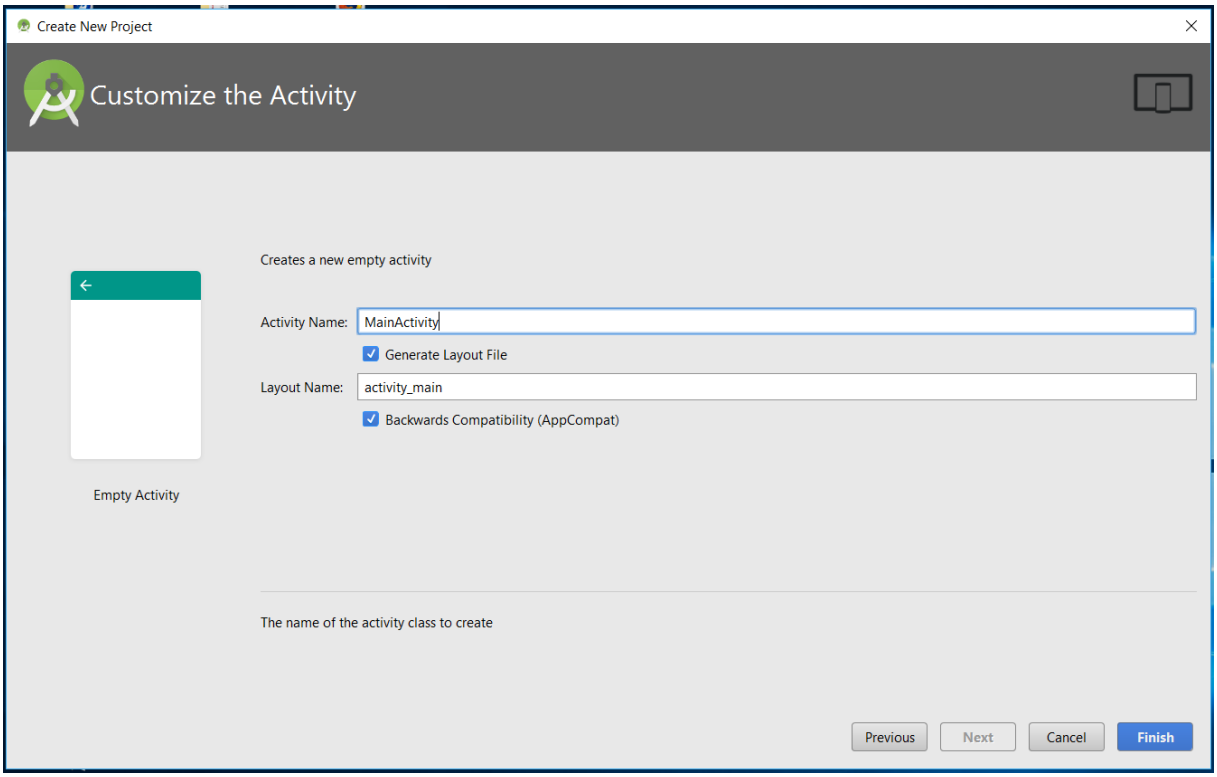
Achte bei der Wahl einer Plattform auf den Haken vor „App und Tablet“. Klicke „Next“.



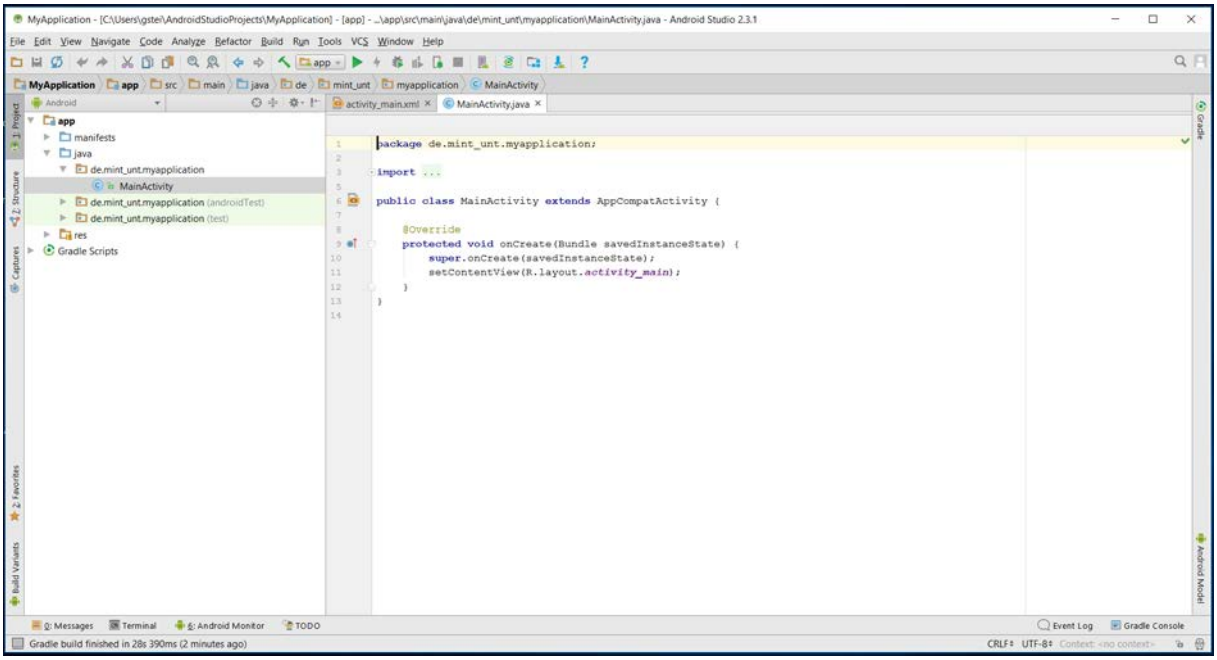
Wähle „Empty Activity“, klicke „Next“.



Lass die Einstellungen im nächsten Fenster unverändert und klicke „Finish“.

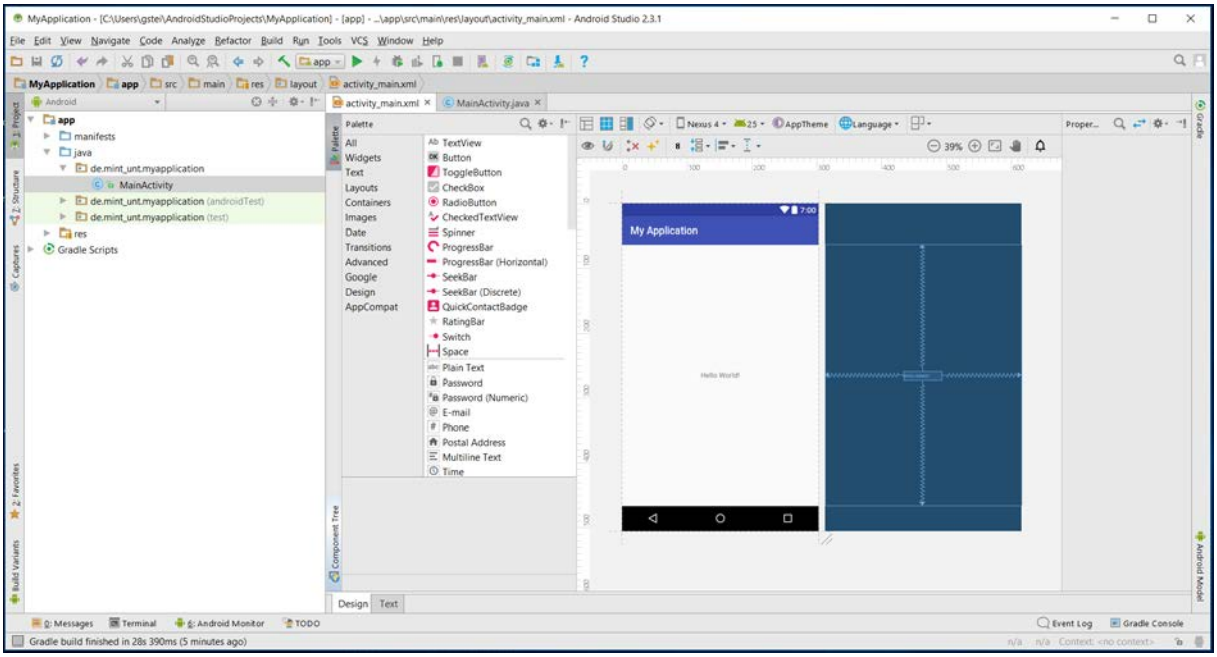


Startbildschirm:

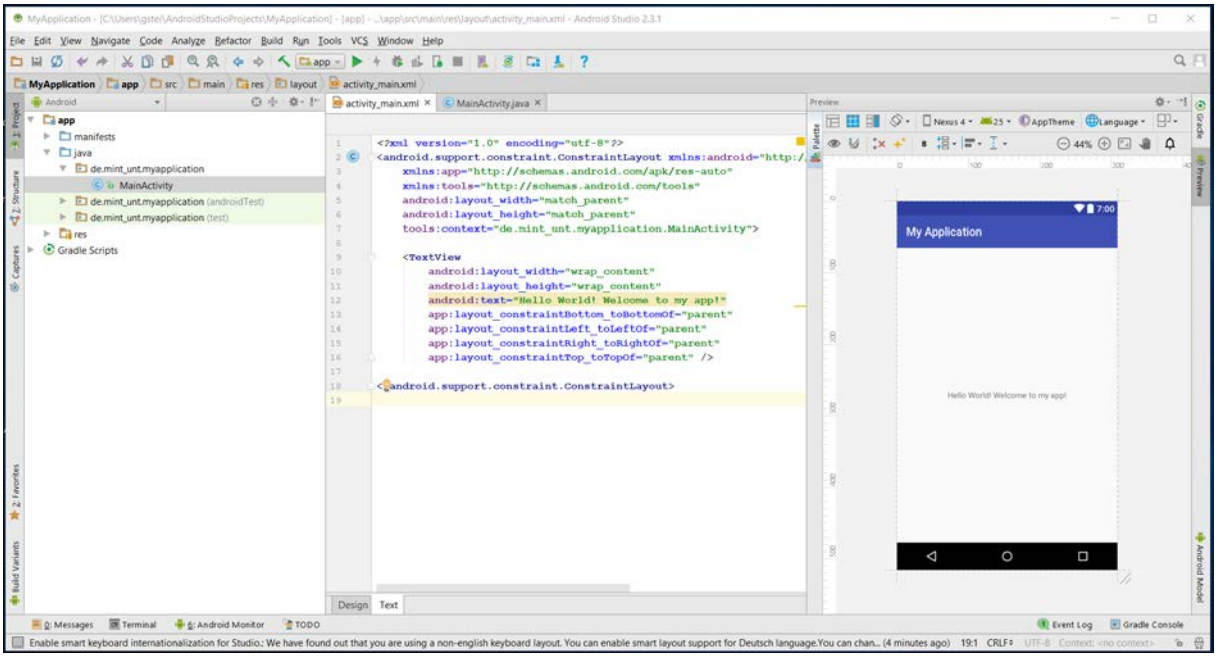


Stufe 2: Die Willkommens-Anzeige verändern

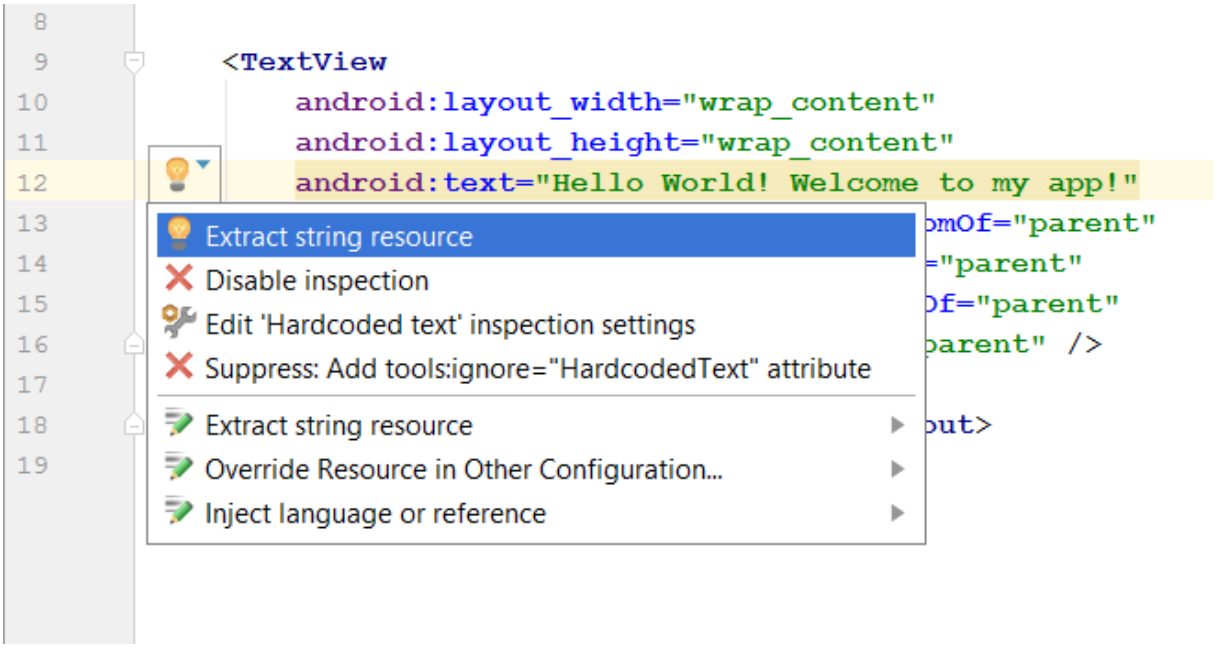
Wähle statt „MainActivity.java“ den Reiter „activity_main.xml“ über der Programmieroberfläche an.



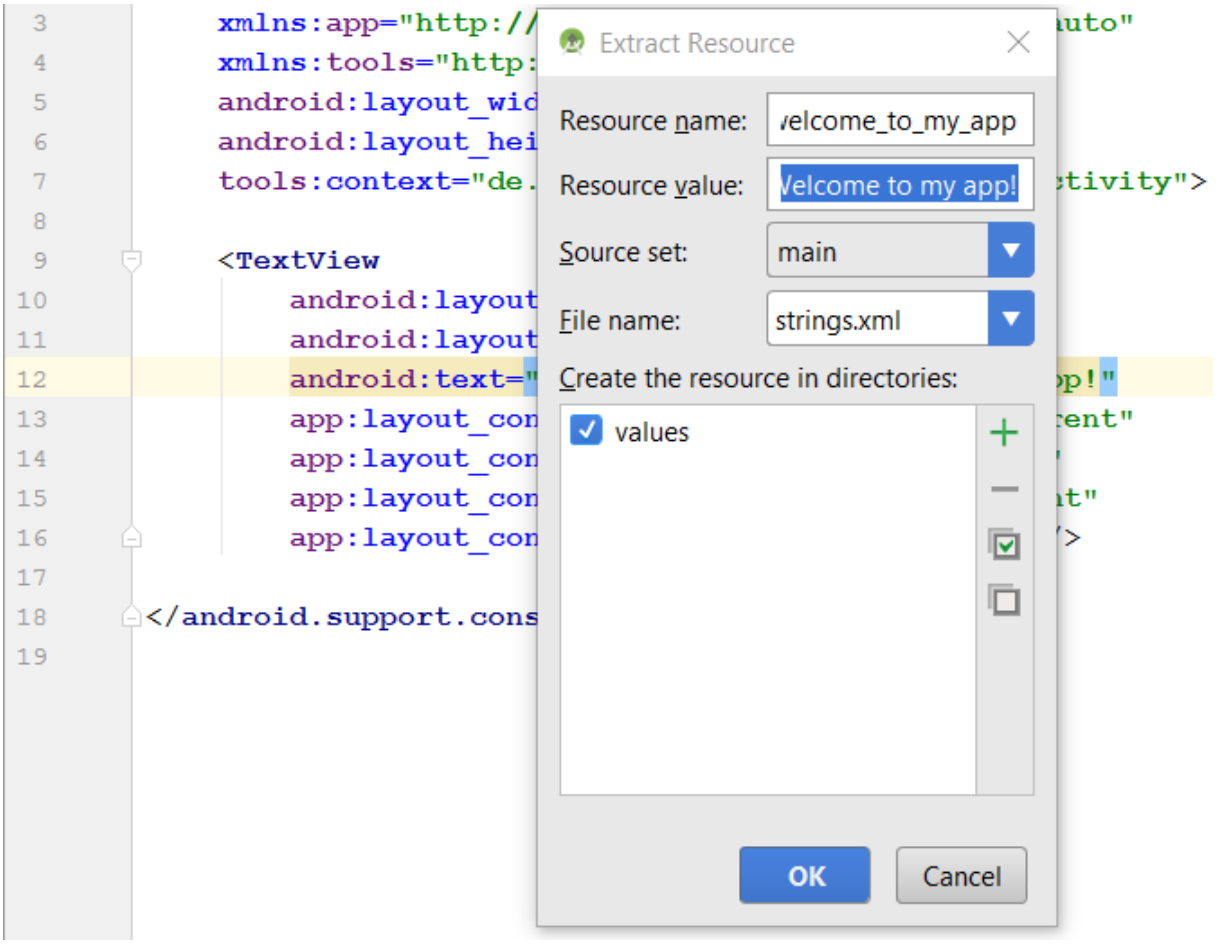
Wähle (unten links) statt „Design“ den Reiter „Text“ aus und suche unter „<TextView“ nach „Hello World!“. Ergänze dort „Welcome to my app!“



Klick in die veränderte Zeile und führe anschließend den Cursor zu dem auf der linken Seite erscheinenden Symbol (eine Lampe). Wenn sich kurz darauf der Hinweis „Click or press Alt + Eingabe“ zeigt, folge ihm, um folgendes Fenster zu öffnen:



Mit der Eingabe-Taste wird eine weitere Anzeige aktiviert:



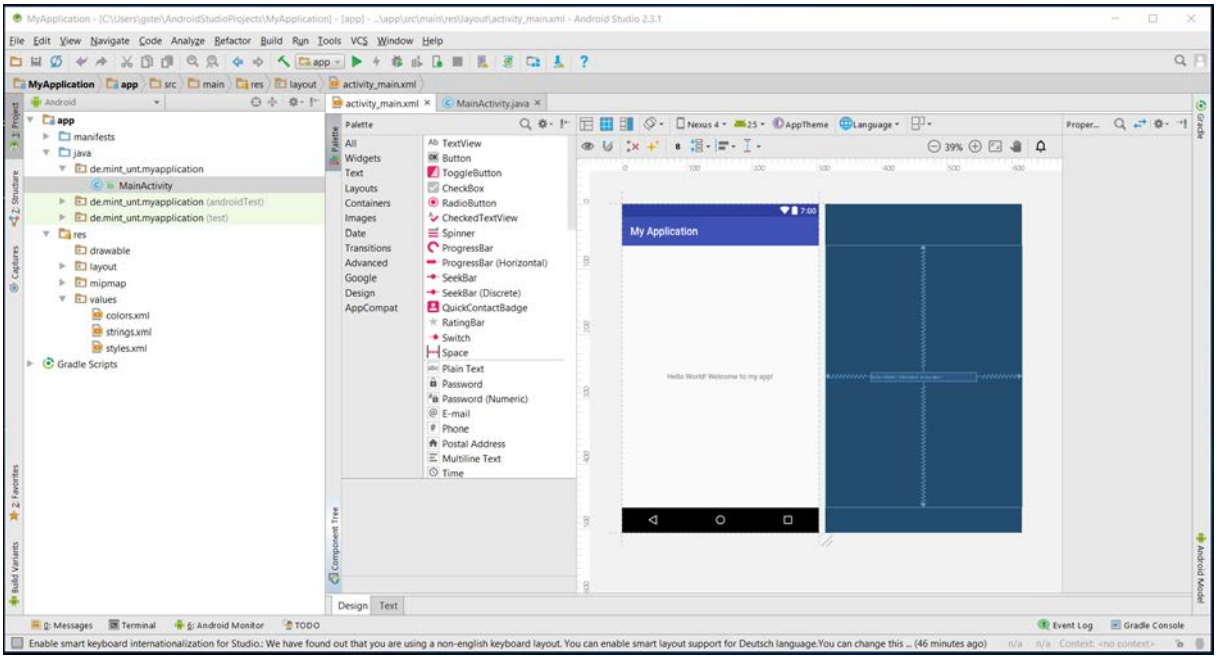
„OK“ erstellt einen „String“ und wandelt die Textzeile „Hello World! Welcome to my app“ in „@string/hello_world_welcome_to_my_app“ um. Die Daten werden (unter >res >values) in der Datei „strings.xml“ gespeichert.

Im Design-Modus lautet der Text in der Mitte des virtuellen Screens nun „Hello World! Welcome to my app!

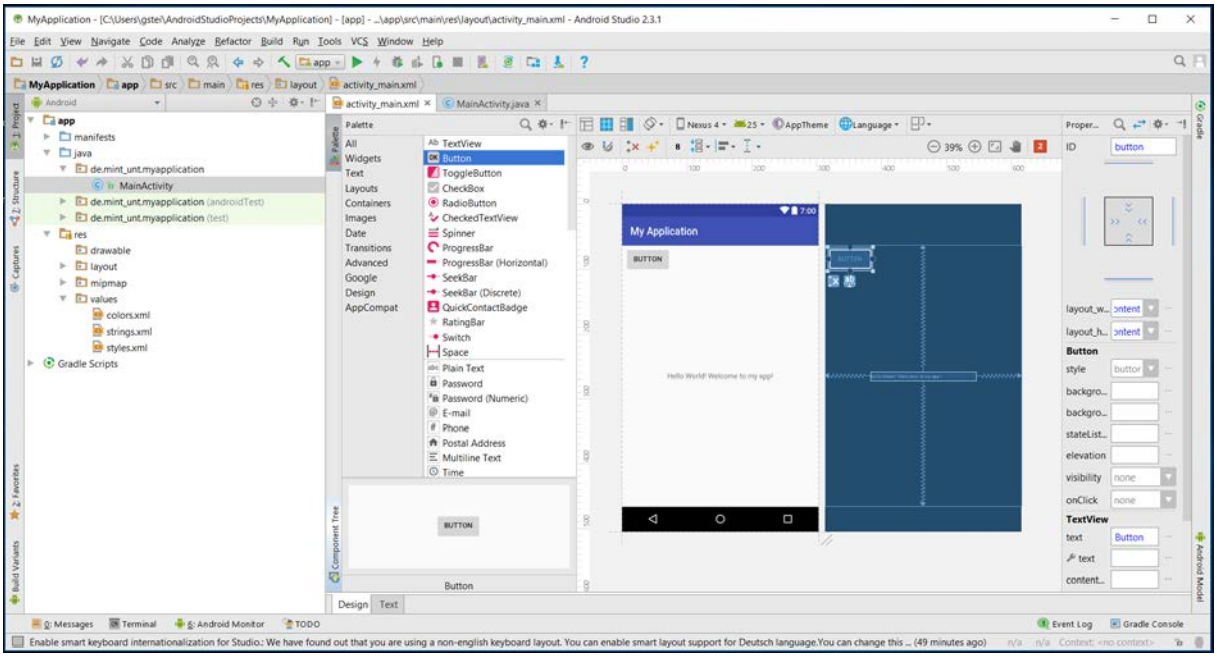
Hinweis: Das Widget „TextView“ ist zum Erstellen jeglicher Form von informellen Texten das richtige Werkzeug.

Stufe 3: Einen Umschalt-Knopf in der „MainActivity“ einrichten

Im Design-Modus der „activity_main.xml“ findet sich links neben der Programmieroberfläche eine „Palette“ von vorgefertigten Objekten.

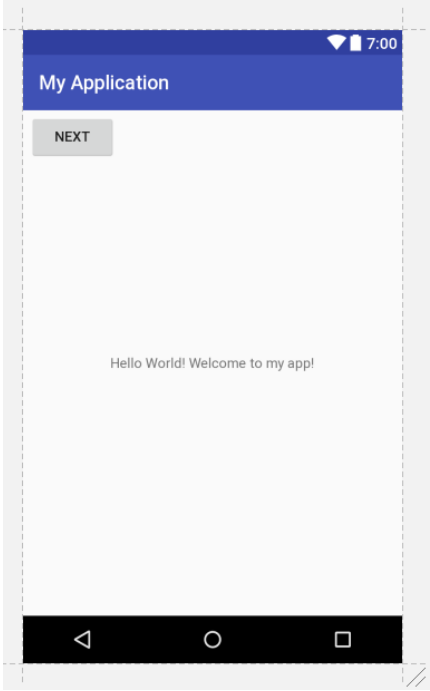


Ziehe aus dieser Palette den „OK Button“ in die linke obere Ecke des virtuellen App-Screens.



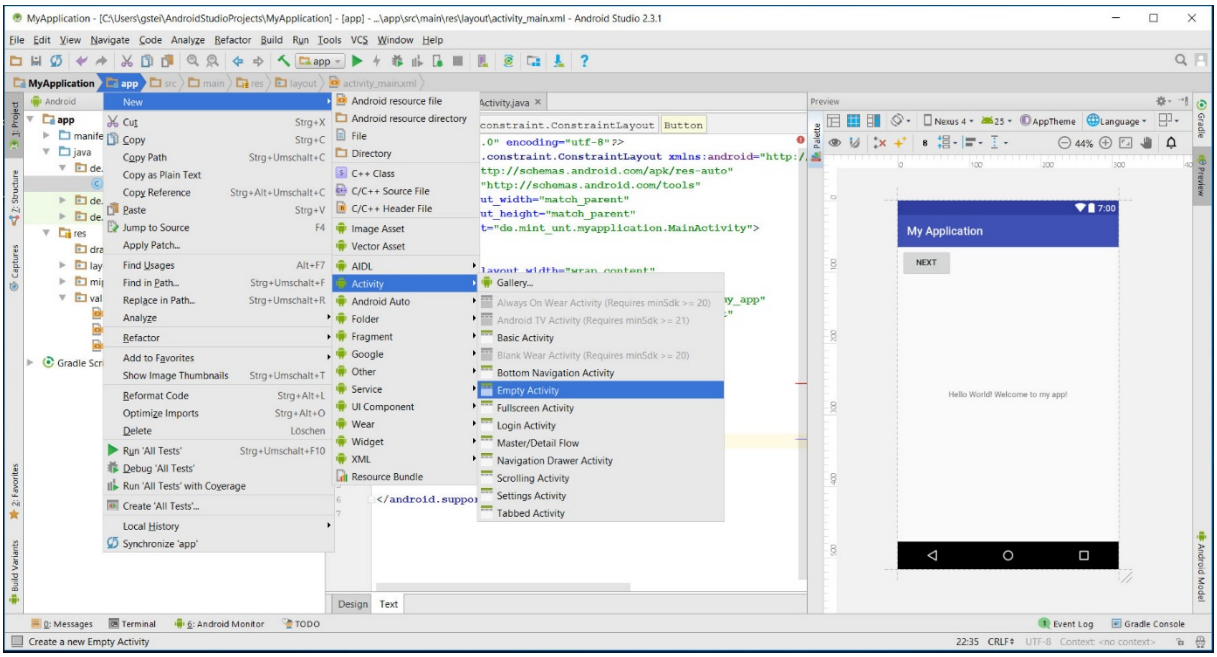
Klicke (unten links) auf den Reiter „Text“ und suche unter „<Button“ nach der Textzeile „Button“. Benenne den Begriff um in „Next“ und erstelle wie gehabt einen weiteren String.

Wenn alles richtiggemacht wurde, steht nun im Design-Modus „NEXT“ in dem kleinen grauen Feld.

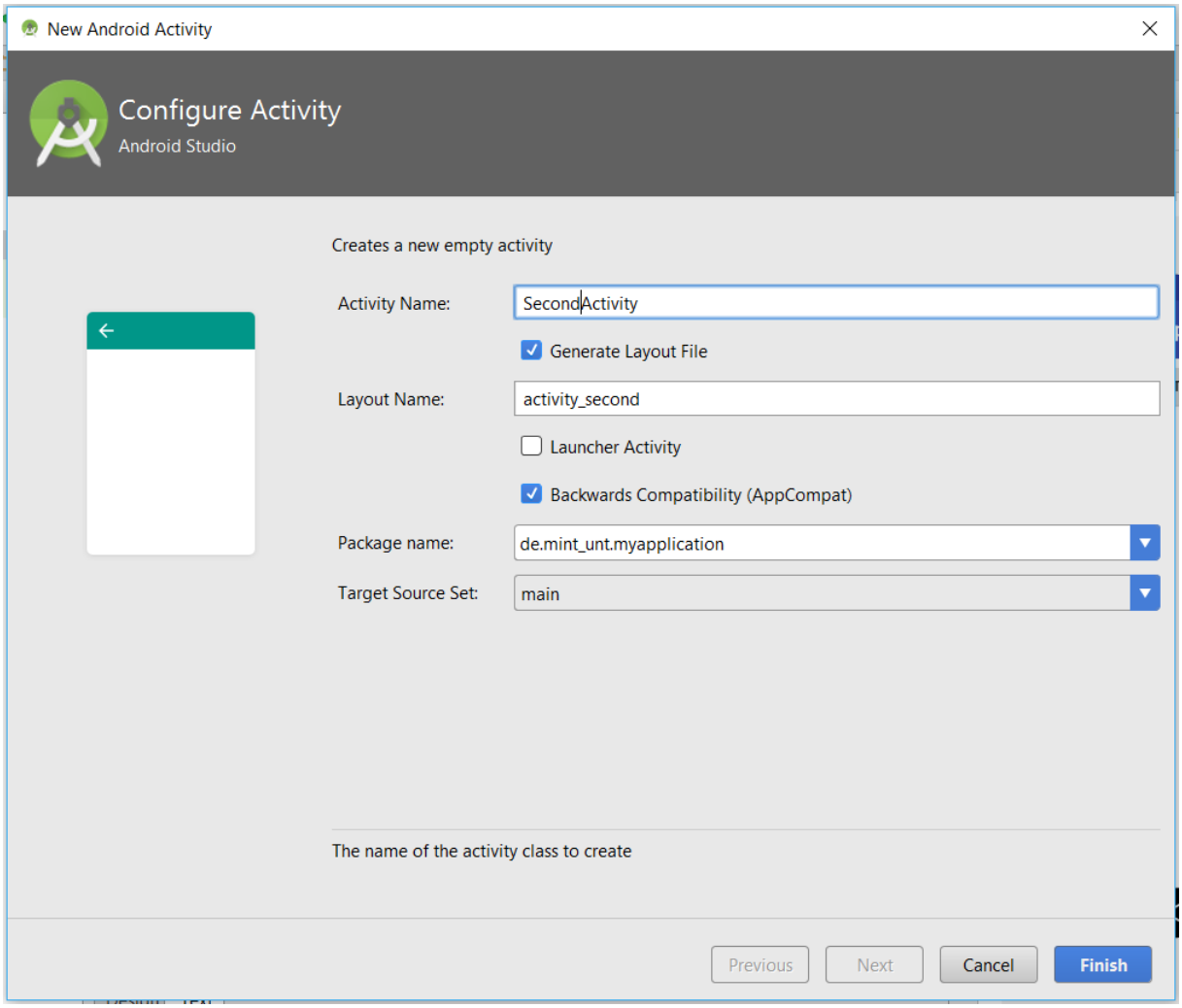


Stufe 4: Eine zweite Seite einrichten

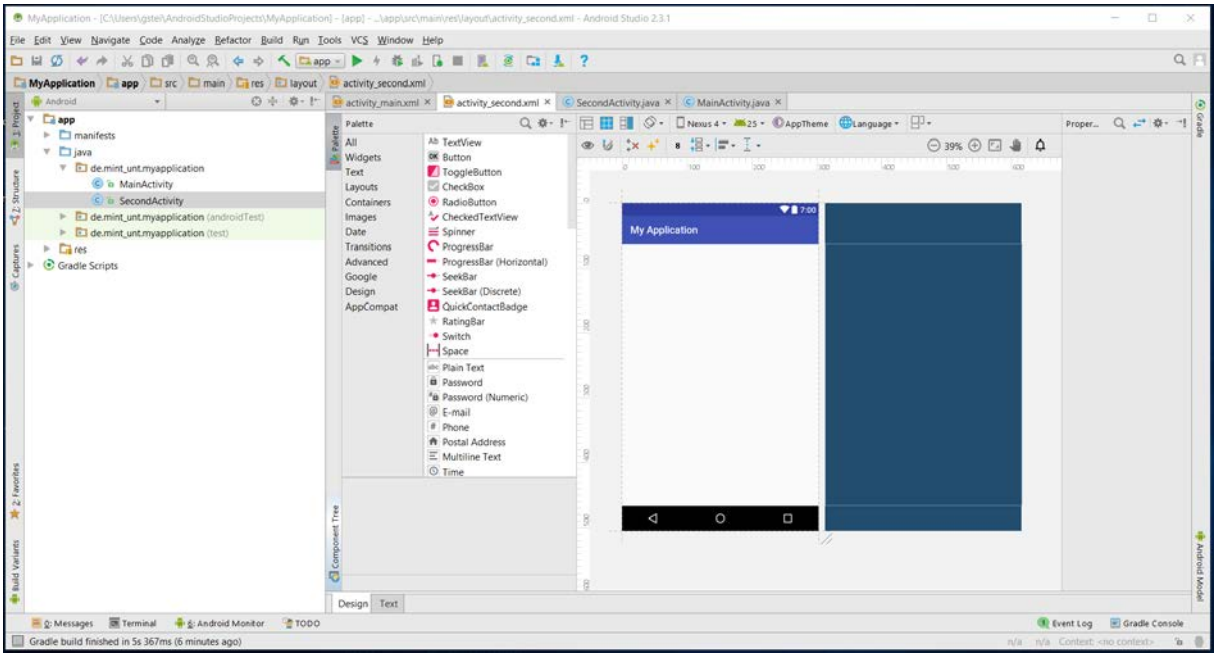
Klicke mit der rechten Maustaste auf „app“ in der Zeile direkt unterhalb des Hauptmenüs und navigiere über >New > Activity zu > EmptyActivity.



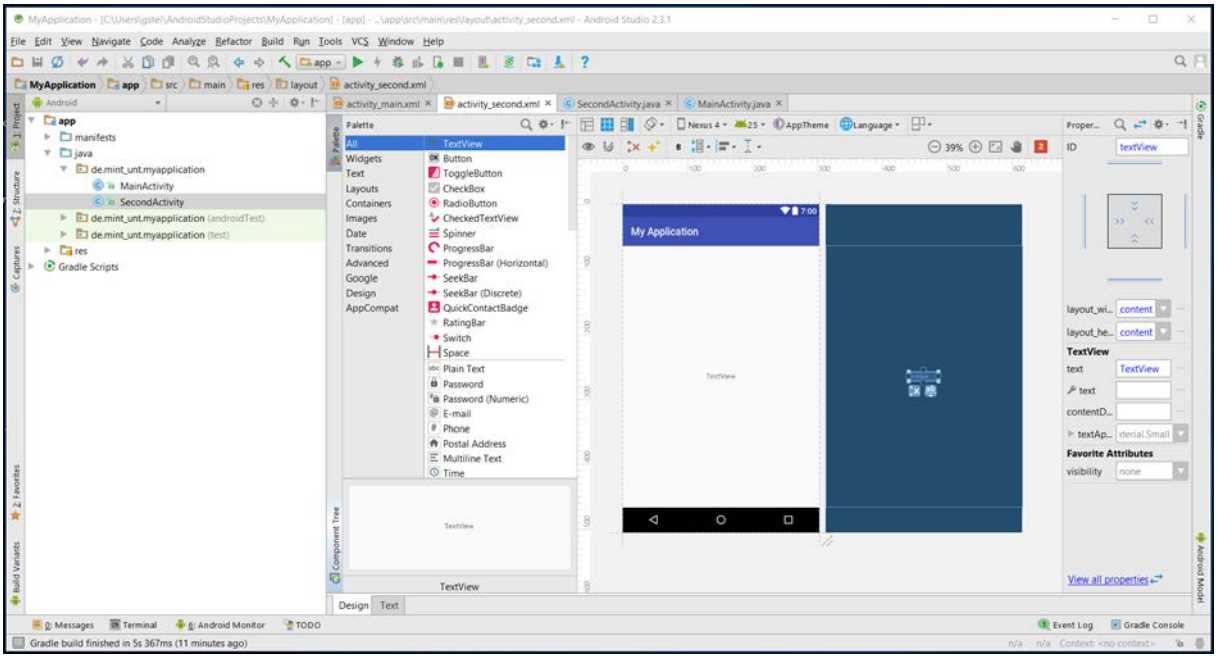
Gib in dem sich öffnenden Fenster der neuen Seite (z.B.) den Namen „SecondActivity“ und klicke auf „Finish“.



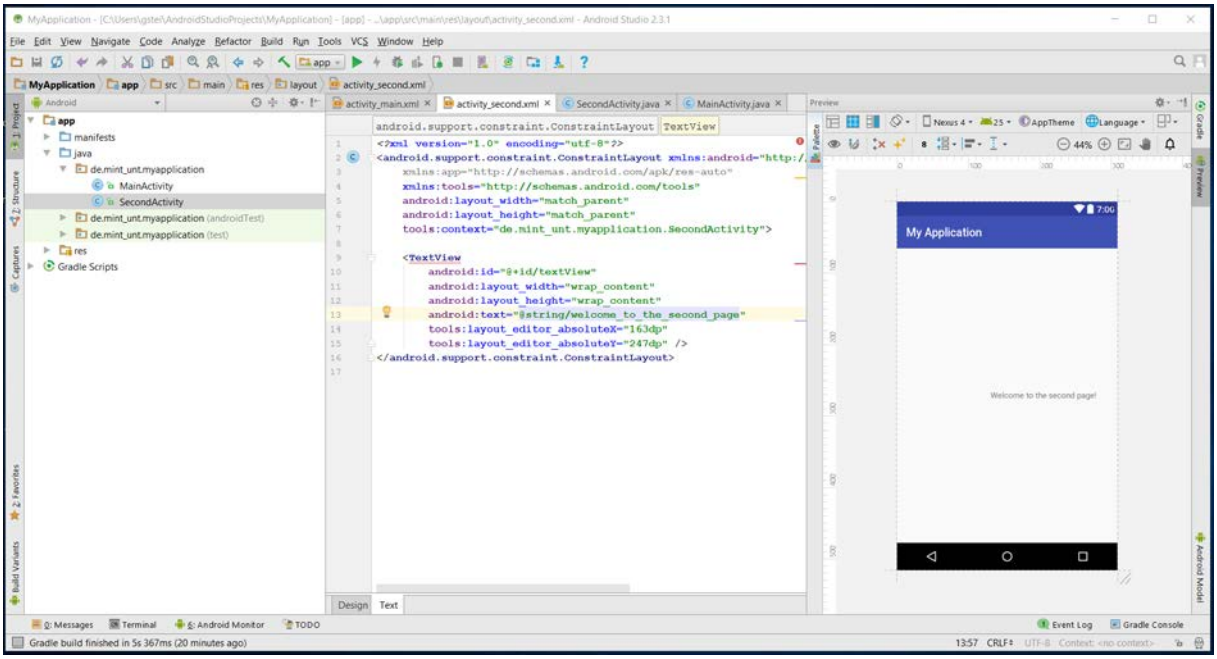
Über der Programmeroberfläche werden zwei neue Dateien angezeigt: „SecondActivity.java“ und „activity_second.xml“. Wähle letztere aus und schalte in die Design-Ansicht.



Ziehe aus der Palette das Objekt „Ab TextView“ in die Mitte des virtuellen App-Screens.



Schalte wieder über den Reiter unten links um in die Textansicht, wandele „TextView“ um in „Welcome to the second page!“ und erstelle wie oben erläutert einen weiteren String. Auf der zweiten Seite steht nun der gewünschte Text.



Stufe 5: Aktivieren des Schaltknopfs

Wähle den „MainActivity.java“ Tabulator und ergänze folgende Zeilen zwischen der letzten Textzeile und den beiden geschweiften Klammern:

```
Button button = (Button) findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                int ce = v.getId();

                if(ce == R.id.button){
                    Intent intent = new Intent(MainActivity.this, SecondActivity.class);
                    startActivity(intent);
                }
```

```
        }
    });
```

Zusätzlich müssen weiter oben im Programmcode folgenden Import-Funktionen ergänzt

```
import android.view.View;
import android.widget.Button;
```

und danach im Code nach rot eingefärbten Begriffen suchen, in diesem Fall „Intend“. Setze mit einem Klick den Cursor auf einen der beiden eingefärbten Begriffe und betätige die Tastenkombination „Alt + Entfernen“. Die Anpassung erfolgt automatisch.

Hier das vollständige Java-Script:

activity_main.xml | activity_second.xml | SecondActivity.java | MainActivity.java

MainActivity

```
1  package de.mint_unt.myapplication;
2
3  import android.content.Intent;
4  import android.support.v7.app.AppCompatActivity;
5  import android.os.Bundle;
6  import android.view.View;
7  import android.widget.Button;
8
9  public class MainActivity extends AppCompatActivity {
10
11     @Override
12     protected void onCreate(Bundle savedInstanceState) {
13         super.onCreate(savedInstanceState);
14         setContentView(R.layout.activity_main);
15         Button button = (Button) findViewById(R.id.button);
16         button.setOnClickListener(new View.OnClickListener() {
17             @Override
18             public void onClick(View v) {
19                 int ce = v.getId();
20
21                 if(ce == R.id.button){
22                     Intent intent = new Intent(MainActivity.this, SecondActivity.class);
23                     startActivity(intent);
24                 }
25             }
26         });
27     }
28 }
29
```

Auf der zweiten Seite lassen sich weitere Schaltknöpfe einrichten, mit denen man z.B. zur ersten zurückkehren oder eine Internetverbindung aufrufen kann. Hier der passende Code zum Script „SecondActivity.java“:

```
package de.mint_unt.myapplication;

import android.content.Intent;
import android.net.Uri;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class SecondActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second);
        Button button = (Button) findViewById(R.id.button3);
        //Button button2 = (Button) findViewById(R.id.button2);
        button.setOnClickListener(new View.OnClickListener() {
```

```
            @Override
            public void onClick(View view)
            {
                int ce = view.getId();
                    if (ce == R.id.button3) {
                    Intent intent = new Intent(Intent.ACTION_VIEW,
                            Uri.parse("http://www.mint-unt.de"));
                    startActivity(intent);
                    }
                }
        });

        Button button1 = (Button) findViewById(R.id.button2);
        button1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                int cc = v.getId();

                if(cc == R.id.button2){
                    Intent intent = new Intent(SecondActivity.this, MainActivity.class);
                    startActivity(intent);
                }
            }
        });

    }
}
```

Stufe 6: Hintergrundfarben ändern

- Zum Ändern der Hintergrundfarbe bei den **Schaltknöpfen** im Code der jeweiligen Seite unter „<Button“ folgende Zeile ergänzen: „android:background=“#FFFFFF“. Die HTML-Codes der Farben finden sich im Internet. #FFFFFF ist ein sehr helles Grau (grey100, fast weiß).
- Zum Ändern der Hintergrundfarbe des **App-Screens** wird die gleiche Befehlszeile unter „<android.support“ eingefügt.

Stufe 7: Eigene Bilder zu platzieren

Zur Bezeichnung von Bilddateien dürfen nur Unterstriche, Kleinbuchstaben und Zahlen verwendet werden. In der Design-Ansicht kann über >images ein >imageView auf dem virtuellen App-Screen platziert werden. In dem sich öffnenden Dialog kann das Bild ausgewählt werden, sofern es unter dem voreingestellten Pfad gespeichert wurde.

Speicherort: C:\Users\gstei\AndroidStudioProjects\MyApplication\app\src\main\res\drawable

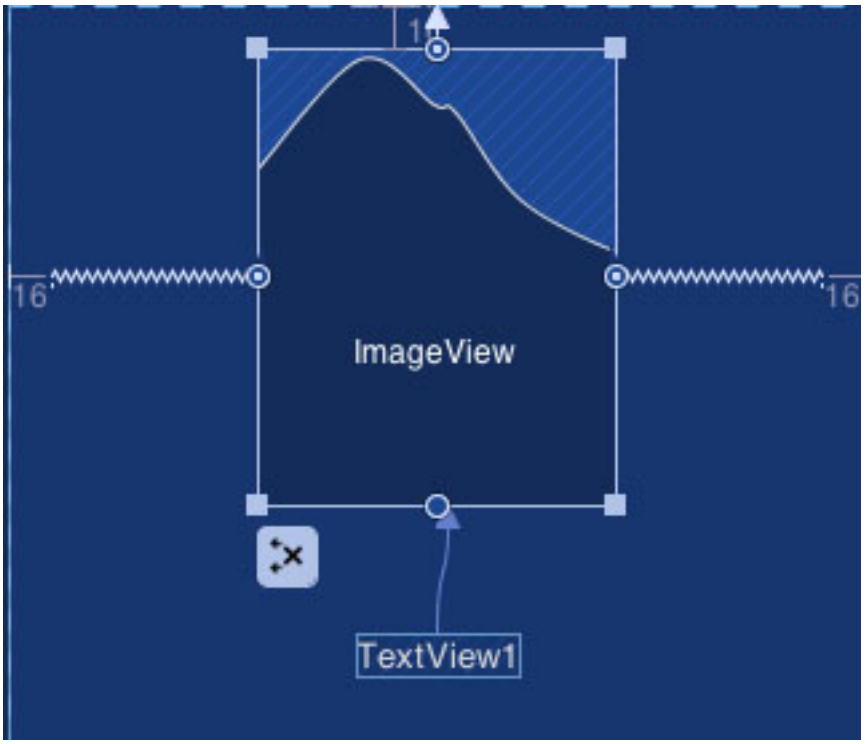
Zum Speichern von Icons funktioniert folgender Weg:

1. Rechtsklick auf res, new **Image Asset**
2. In **Icon Type** wähle **Action Bar and Tab Icons**
3. In **Asset type** wähle **Image**
4. In **Path** wähle den Pfad zum Bild auf deinem Rechner
5. Next->Finish

Stufe 7: Objekte auf dem App-Screen verankern.

Klickt man im Design-Modus einen Text oder Schaltknopf im an, werden an den Ecken blaue Quadrate (zum Vergrößern) und auf den Kantenmitten Kreise sichtbar. Diese sind zum Platzieren relativ zu anderen Objekten oder den Bildschirmrändern gedacht. Zieht man einen solchen Kreis zum Bildschirmrand, folgt das Objekt nach. Zieht man danach den auf dem Objekt gegenüberliegenden Kreis zum gegenüberliegenden Bildschirmrand, wird das Objekt auf dieser Achse zentriert usw.

Welche Fixierungen eingerichtet worden sind, wird auf dem blauen Screen mit Zick-zack-Linien angezeigt.



Objekte platziert man relativ zueinander, indem man an passenden Seiten die Kreise übereinander zieht. Ein rotes Aufleuchten zeigt die Verbindung an.

Löschen kann man die Verbindungen durch einen Klick im blauen Screen auf das Kreuz links unterhalb des Objekts, das beim Anwählen eingeblendet wird. Näheres findet sich hier.