Android Apps entwickeln

Eine Einführung

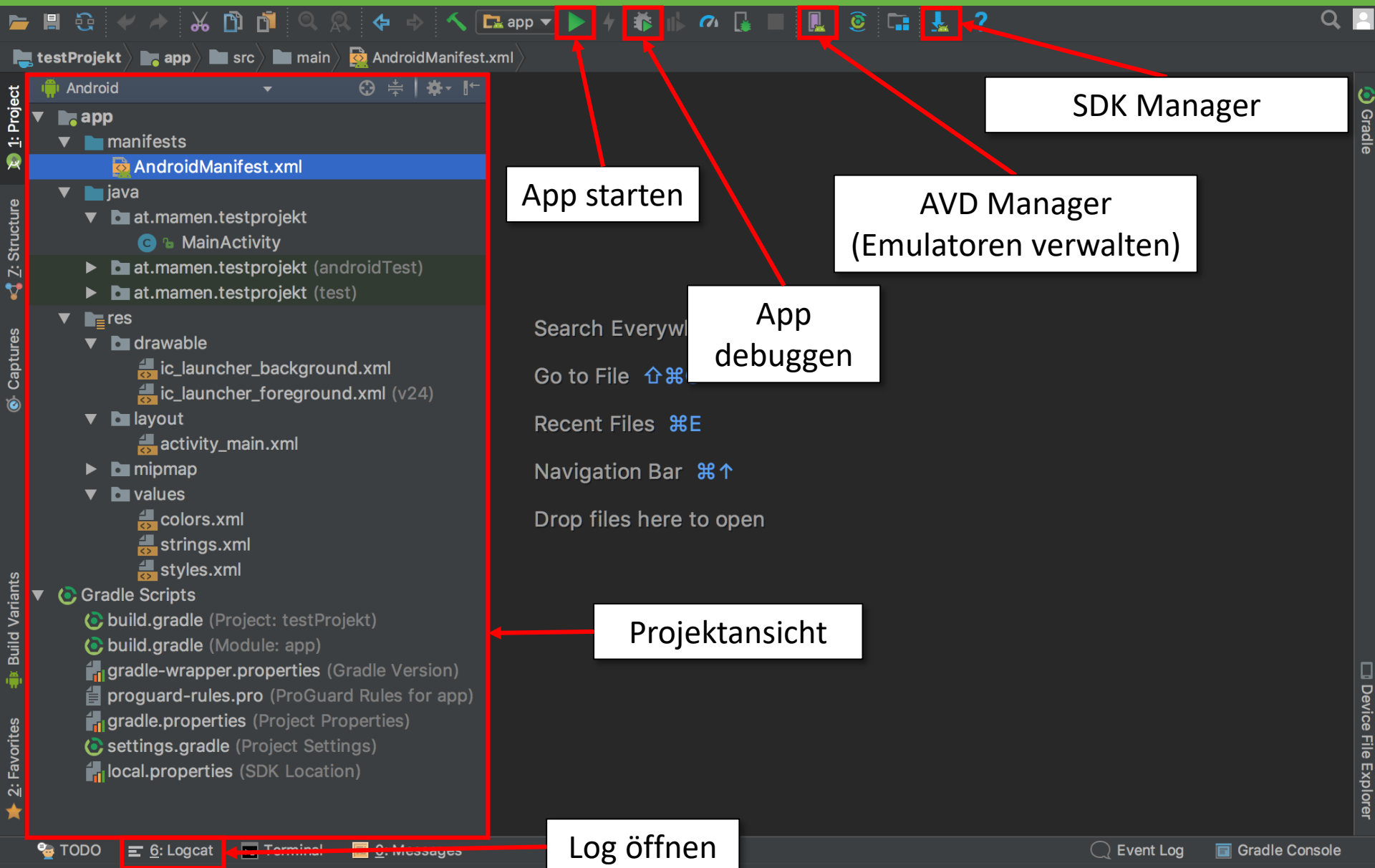
Alexander Lercher

Folien von Markus Mennel

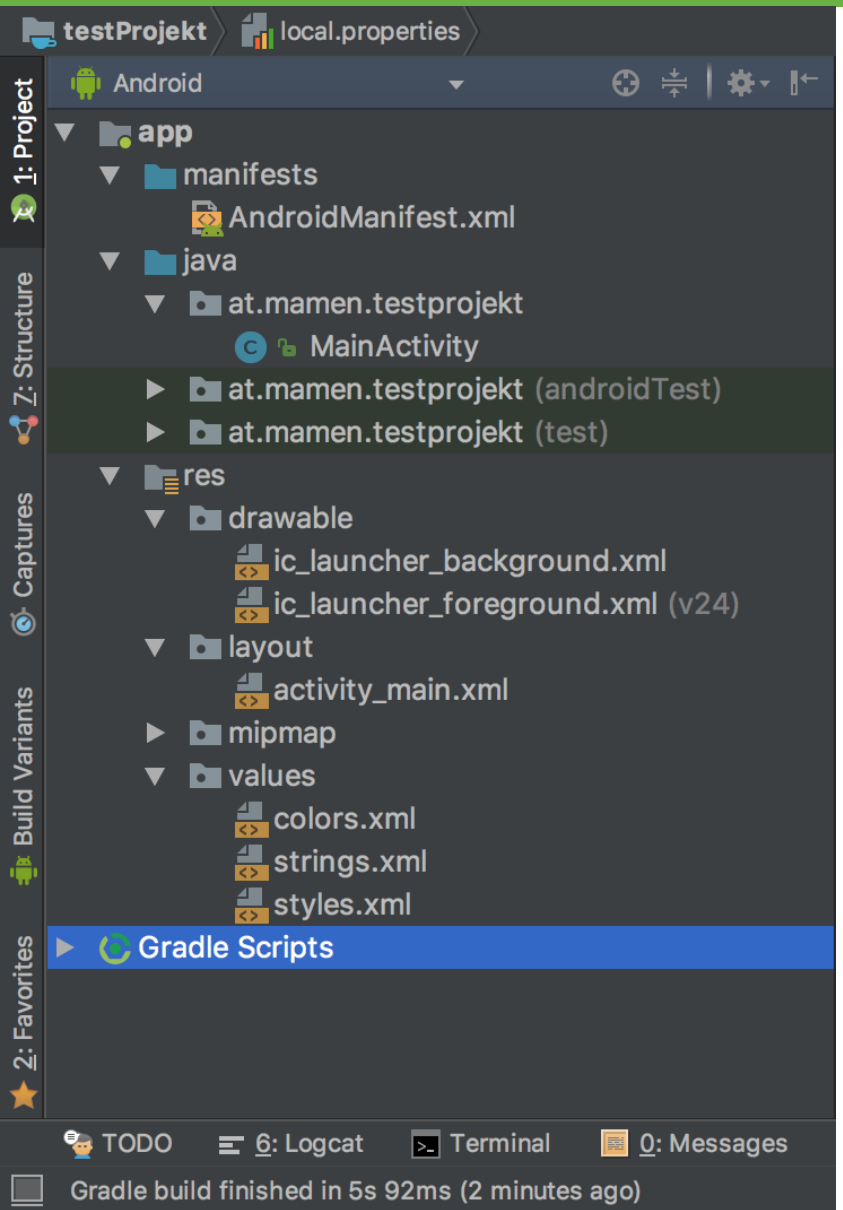
Inhalt

- Android Studio UI
- Aufbau einer App
- View erstellen
- Code mit View verknüpfen
- App starten / debuggen
- Logging

Android Studio UI

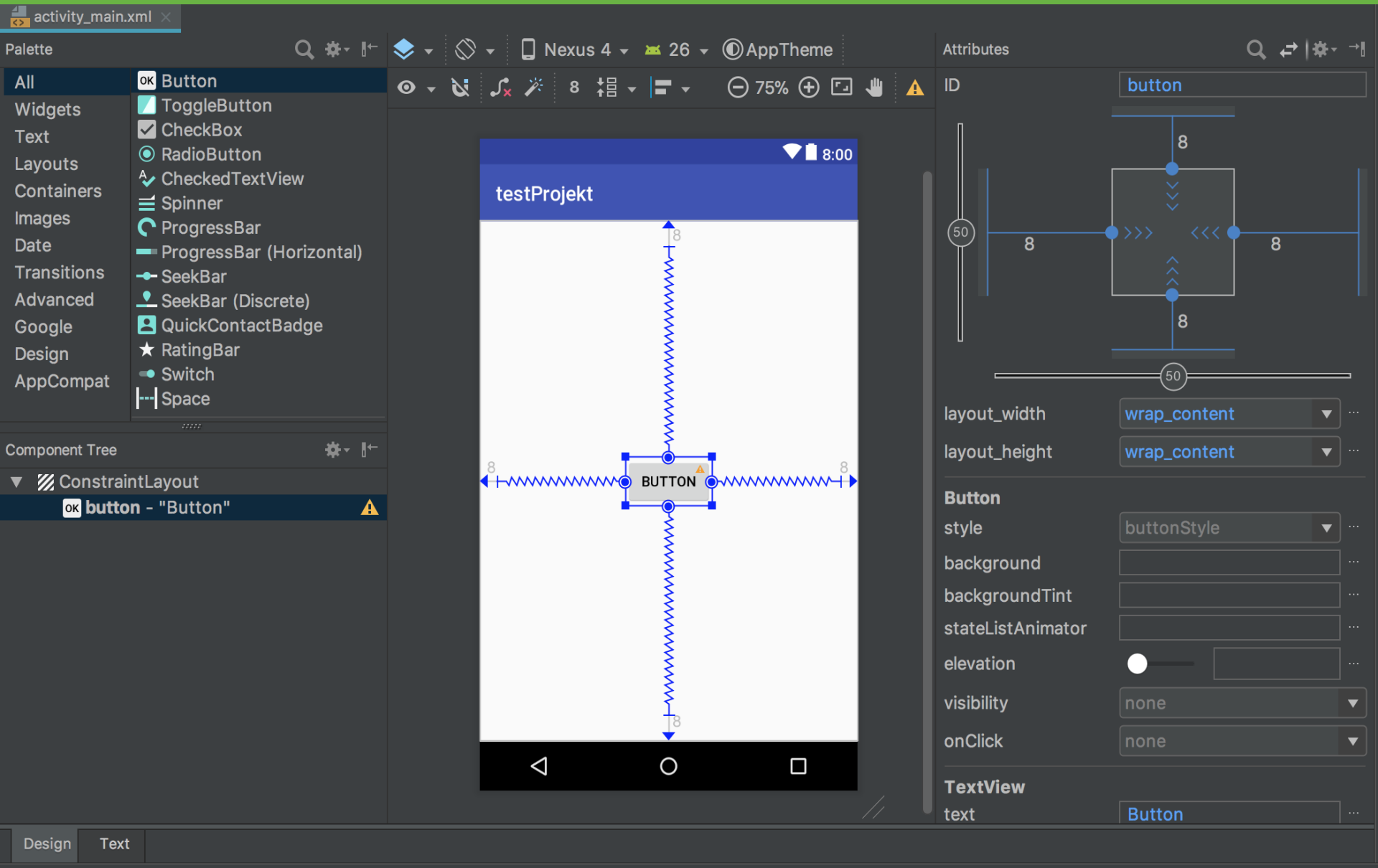


Aufbau einer App

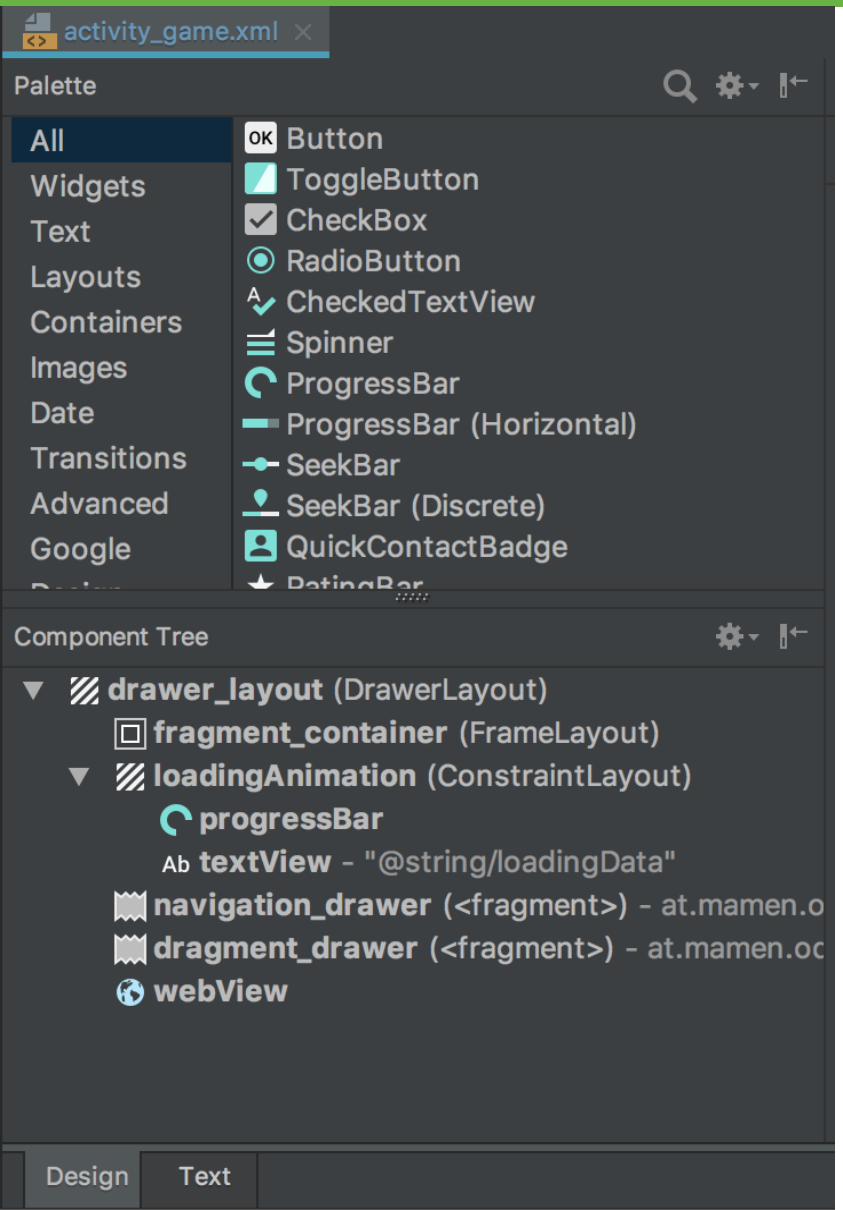


- manifest
  - beinhaltet Informationen über App (z.B. Name, Activities etc.)
- java
  - Sourcecode
  - Tests
- res
  - Drawable: Bilder
  - Layout: UI-Layouts
  - Mipmap: App-Icon
  - Values: Farben & Strings
- Gradle Scripts
  - Definition der min. SDK etc.

View erstellen

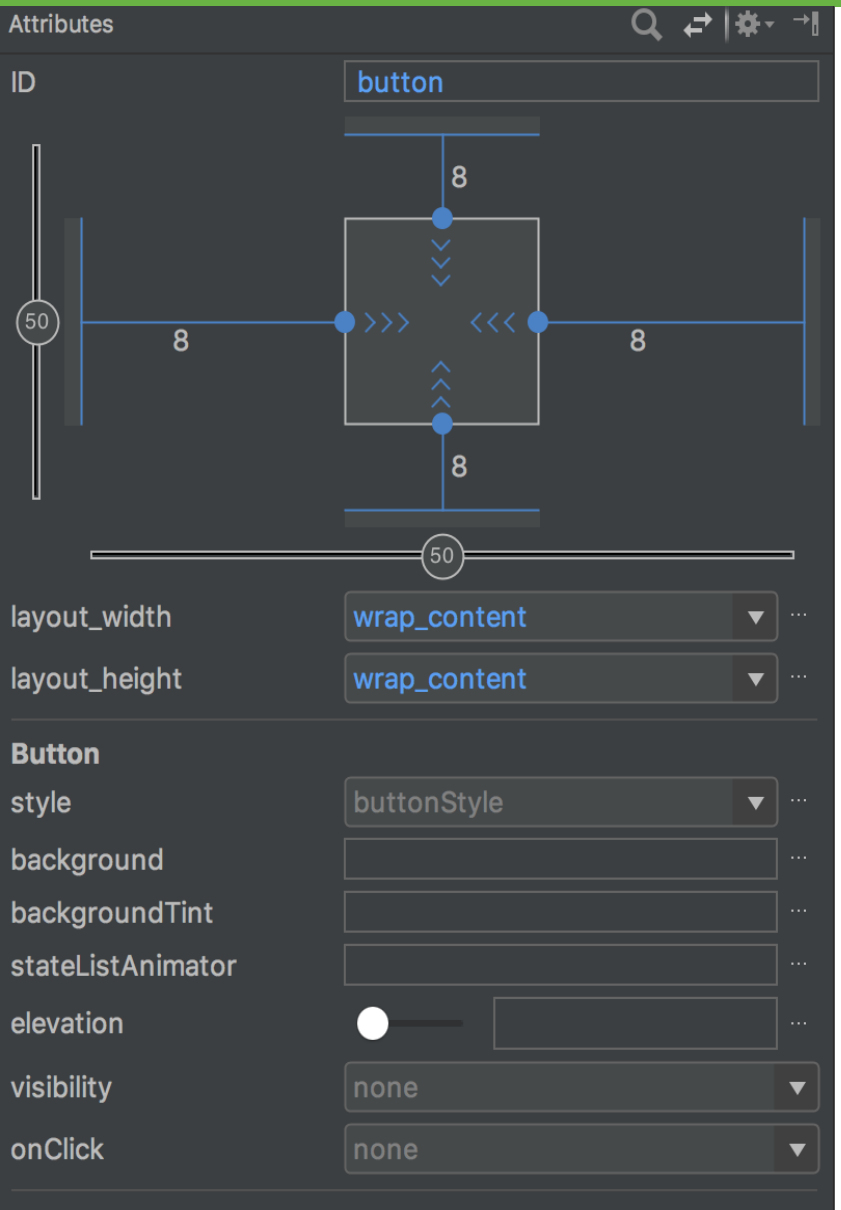


View erstellen



- Palette
  - Enthält alle verfügbaren UI-Elemente (Buttons, Textfelder etc.)
  - Elemente werden mittels Drag & Drop zum View hinzugefügt
  - Im Hintergrund wird das Layout als XML-Datei laufend angepasst
- Component Tree
  - Veranschaulicht alle verwendeten UI-Elemente und deren Verschachtelung

View erstellen



- Attributes
  - Bei einem Klick auf ein platziertes Element werden die Attribute ersichtlich
  - Constraints, Aussehen und Sichtbarkeit können hier verändert werden
  - Die vergebene ID wird später beim Programmieren benötigt und **muss eindeutig** sein

View erstellen

activity_main.xml

android.support.constraint.ConstraintLayout

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <android.support.constraint.ConstraintLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:app="http://schemas.android.com/apk/res-auto"
5      xmlns:tools="http://schemas.android.com/tools"
6      android:layout_width="match_parent"
7      android:layout_height="match_parent"
8      tools:context="at.mamen.testprojekt.MainActivity">
9
10     <Button
11         android:id="@+id/button"
12         android:layout_width="wrap_content"
13         android:layout_height="wrap_content"
14         android:layout_marginBottom="8dp"
15         android:layout_marginEnd="8dp"
16         android:layout_marginStart="8dp"
17         android:layout_marginTop="8dp"
18         android:text="Button"
19         app:layout_constraintBottom_toBottomOf="parent"
20         app:layout_constraintEnd_toEndOf="parent"
21         app:layout_constraintStart_toStartOf="parent"
22         app:layout_constraintTop_toTopOf="parent" />
23
24 </android.support.constraint.ConstraintLayout>
25
```

Design Text

Code mit View verknüpfen

Beispiel: Button

1. Variable definieren
   - Button btn;
2. UI-Element der Variable zuweisen
   - btn = findViewByID(R.id.buttonID);
3. Klick-Eventhandler setzen
   - btn.setOnClickListener(
     new View.OnClickListener() { … }
     );

Code mit View verknüpfen

Beispiel: Textfeld

1. Variable definieren
   - EditText txt;
2. UI-Element der Variable zuweisen
   - txt = findViewByID(R.id.editTextID);
3. Text auslesen
   - String text = txt.getText().toString();

App starten / debuggen

- Die App kann entweder im Emulator oder auf dem eigenen Smartphone gestartet werden
- Ein Emulator kann im AVD-Manager erstellt werden
- Der Emulator funktioniert **ausschließlich** auf Intel-Prozessoren und ist im Vergleich zum eigenen Smartphone sehr langsam
- Nicht alle Funktionen sind im Emulator verwendbar (z.B. Kommunikation mit Geräten im selben Netzwerk)
- Für die Verwendung des eigenen Smartphones werden die entsprechenden USB-Treiber benötigt

Logging

- In Android wird nicht auf die Konsole geschrieben, sondern in den Logcat.
- Es gibt 5 verschiedene Log-Level:
  - **Debug:** Log.d("TAG","Hallo Welt!");
  - **Error:** Log.e("TAG","Hallo Welt!");
  - **Info:** Log.i("TAG","Hallo Welt!");
  - **Verbose:** Log.v("TAG","Hallo Welt!");
  - **Warn:** Log.w("TAG","Hallo Welt!");
- Der TAG hilft dabei, den Log-Eintrag zu kategorisieren

```
03-04 21:39:19.496 3101-3101/at.mamen.testprojekt D/BeispielTAG: Ich bin ein Beispiellogeintrag mit Log.d()
03-04 21:39:19.496 3101-3101/at.mamen.testprojekt E/BeispielTAG: Ich bin ein Beispiellogeintrag mit Log.e()
```

Fragen?

Stefan.Goenitzer@aau.at

Selene.Lobnig@aau.at

Alexander.Lercher@aau.at