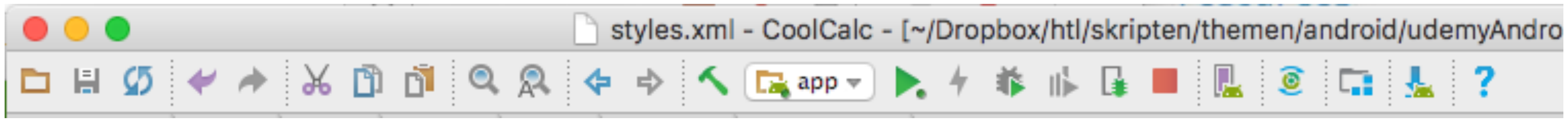
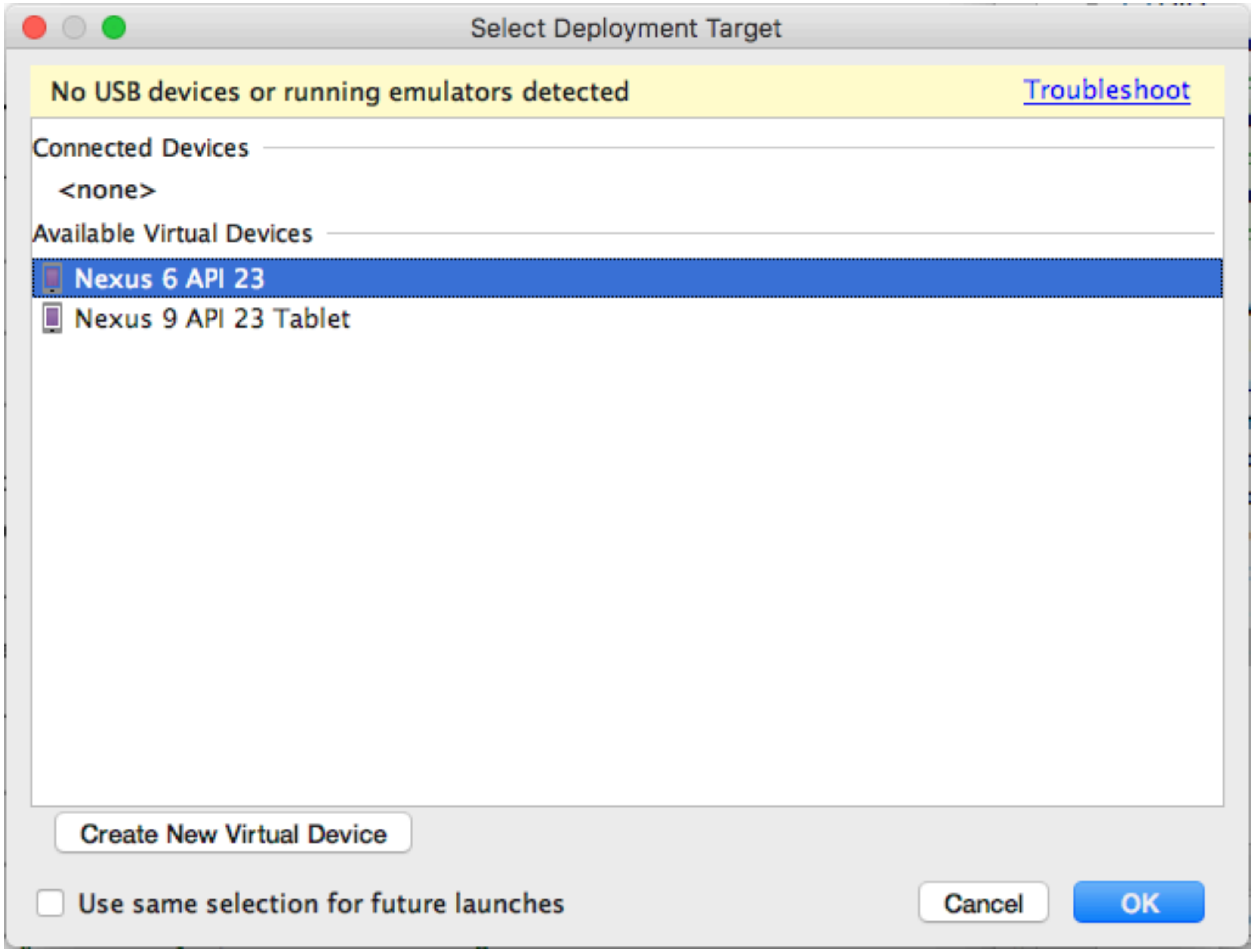
# CoolCalc


styles.xml - CoolCalc - [~/Dropbox/htl/skripten/themen/android/udemyAndro


Select Deployment Target

No USB devices or running emulators detected

Connected Devices

<none>

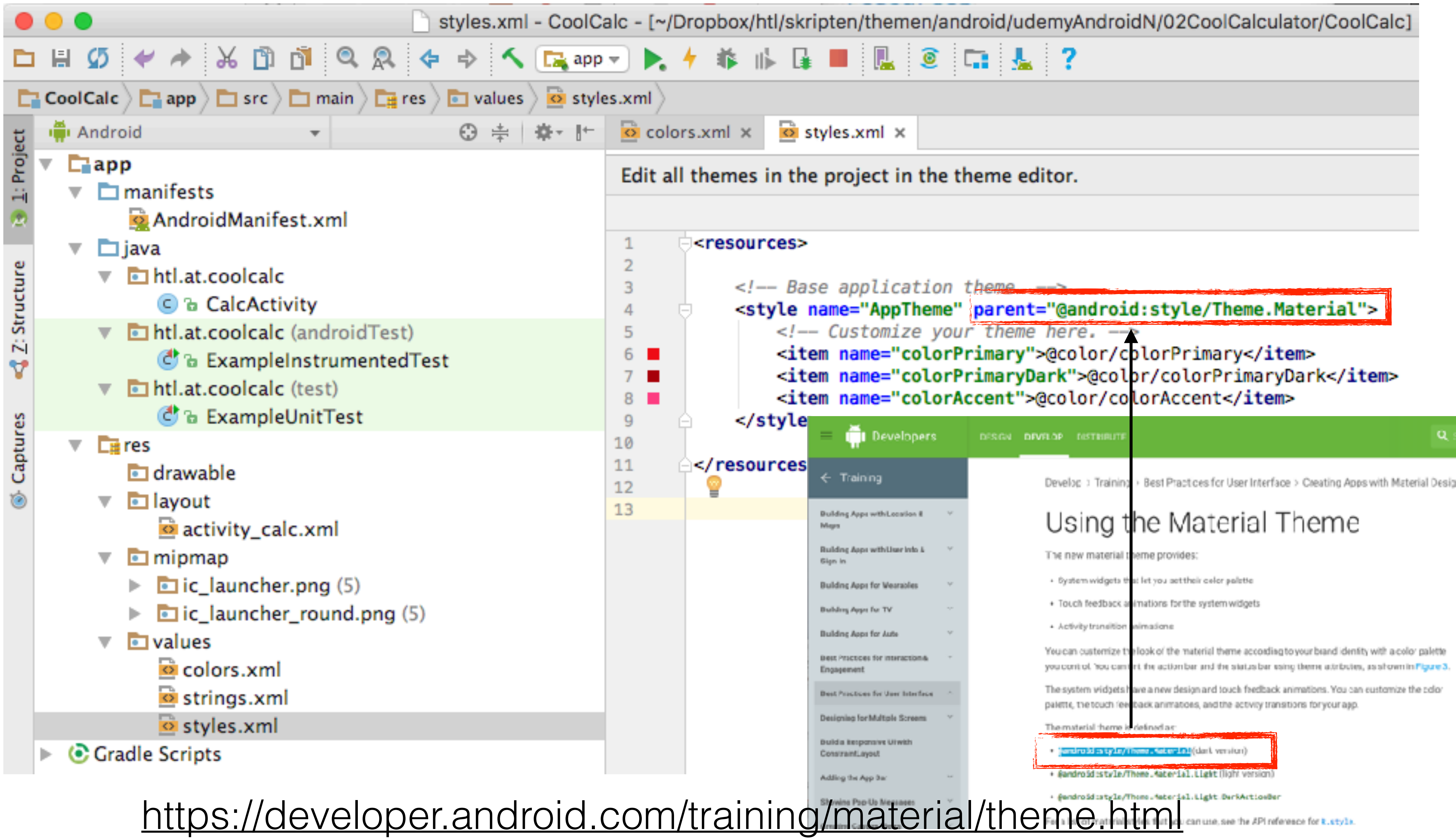
Available Virtual Devices

Nexus 6 API 23

Nexus 9 API 23 Tablet

Use same selection for future launches

# Ändern des Themes auf Material Theme


styles.xml - CoolCalc - [~/Dropbox/htl/skripten/themen/android/udemyAndroidN/02CoolCalculator/CoolCalc]
app
CoolCalc › app › src › main › res › values › styles.xml
1: Project
7: Structure
Captures
Android
colors.xml
styles.xml

- app
  - manifests
    - AndroidManifest.xml
  - java
    - htl.at.coolcalc
      - CalcActivity
    - htl.at.coolcalc (androidTest)
      - ExampleInstrumentedTest
    - htl.at.coolcalc (test)
      - ExampleUnitTest
  - res
    - drawable
    - layout
      - activity_calc.xml
    - mipmap
      - ic_launcher.png (5)
      - ic_launcher_round.png (5)
    - values
      - colors.xml
      - strings.xml
      - styles.xml
- Gradle Scripts

Edit all themes in the project in the theme editor.

```xml
<resources>

    <!-- Base application theme. -->
    <style name="AppTheme" parent="@android:style/Theme.Material">
        <!-- Customize your theme here. -->
        <item name="colorPrimary">@color/colorPrimary</item>
        <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
        <item name="colorAccent">@color/colorAccent</item>
    </style

</resources
```

Developers
DESIGN
DEVELOP
DISTRIBUTE
Training
Building Apps with Location & Maps
Building Apps with User Info & Sign In
Building Apps for Wearables
Building Apps for TV
Building Apps for Auto
Best Practices for Interaction & Engagement
Best Practices for User Interface
Designing for Multiple Screens
Build a Responsive UI with ConstraintLayout
Adding the App Bar
Showing Pop-Up Messages

Develop › Training › Best Practices for User Interface › Creating Apps with Material Desig

## Using the Material Theme

The new material theme provides:

- System widgets that let you set their color palette
- Touch feedback animations for the system widgets
- Activity transition animations

You can customize the look of the material theme according to your brand identity with a color palette you control. You can tint the action bar and the status bar using theme attributes, as shown in Figure 3.

The system widgets have a new design and touch feedback animations. You can customize the color palette, the touch feedback animations, and the activity transitions for your app.

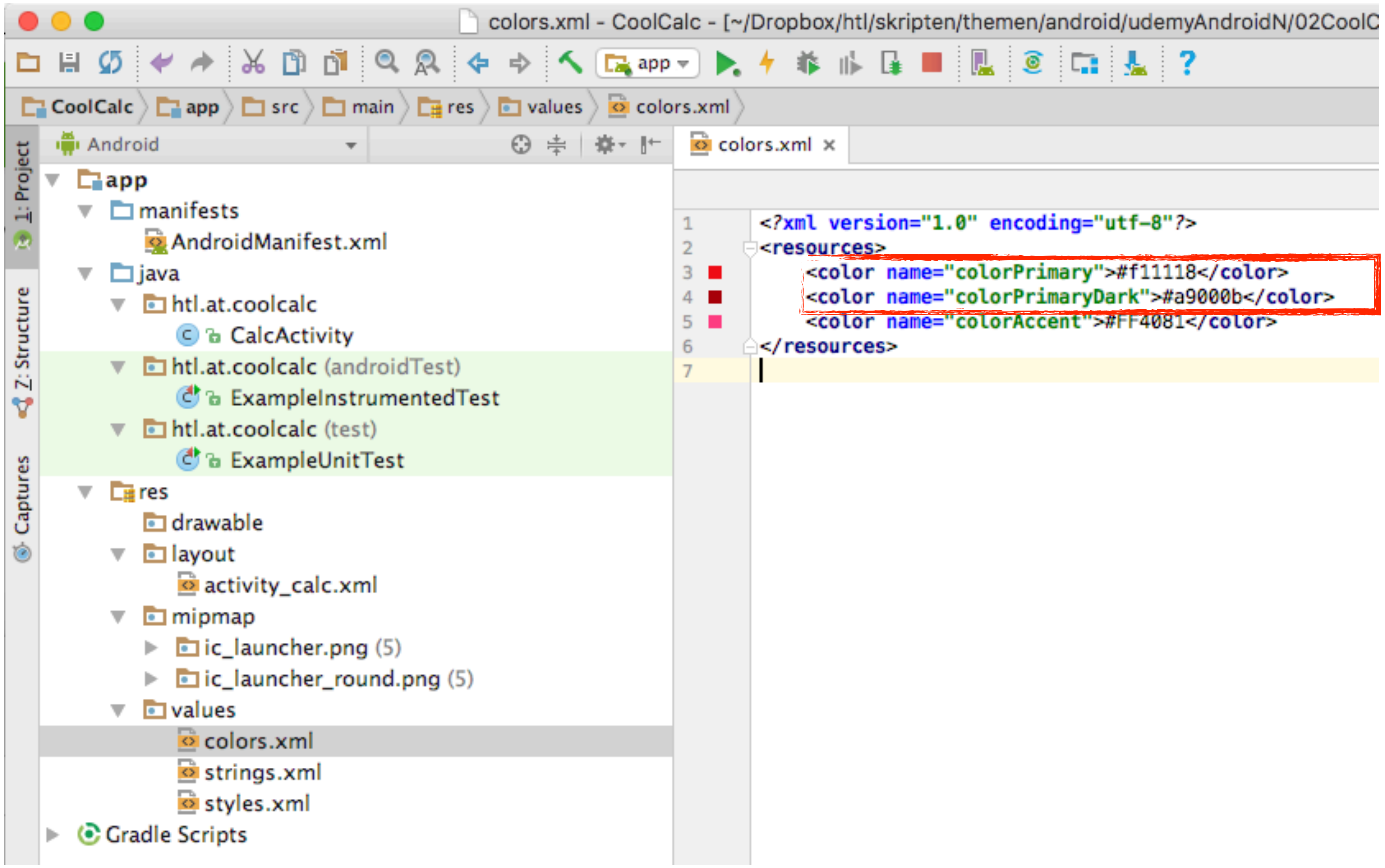
The material theme is defined as:

- @android:style/Theme.Material (dark version)
- @android:style/Theme.Material.Light (light version)
- @android:style/Theme.Material.Light.DarkActionBar

For a list of material styles that you can use, see the API reference for R.style.

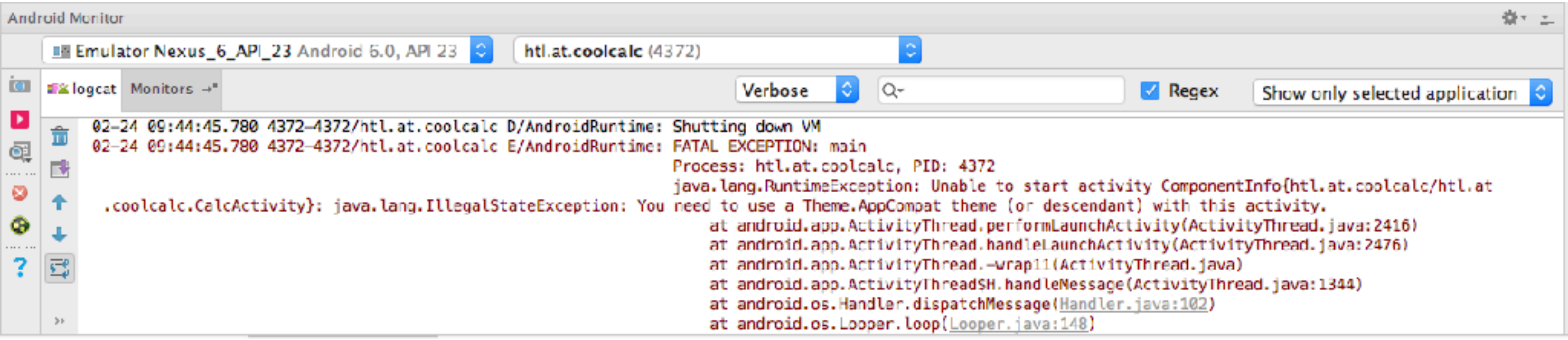
https://developer.android.com/training/material/theme.html

# Ändern der Farbe



```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#f11118</color>
    <color name="colorPrimaryDark">#a9000b</color>
    <color name="colorAccent">#FF4081</color>
</resources>
```

# Fehler beim Ausführen


Android Monitor
Emulator Nexus_6_API_23 Android 6.0, API 23
htl.at.coolcalc (4372)
logcat
Monitors
Verbose
Regex
Show only selected application

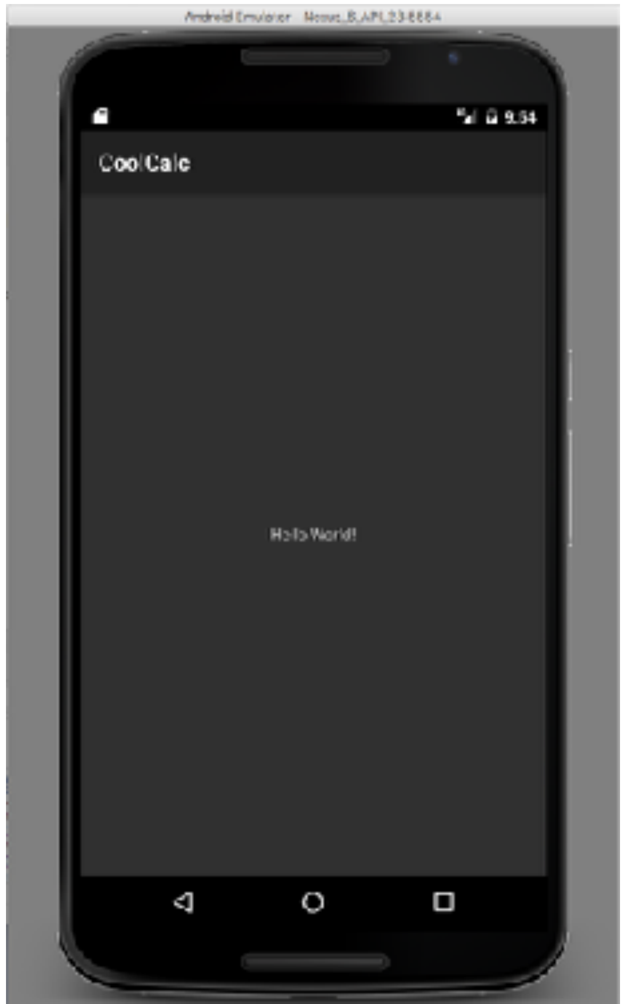
```
02-24 09:44:45.780 4372-4372/htl.at.coolcalc D/AndroidRuntime: Shutting down VM
02-24 09:44:45.780 4372-4372/htl.at.coolcalc E/AndroidRuntime: FATAL EXCEPTION: main
                                                               Process: htl.at.coolcalc, PID: 4372
                                                               java.lang.RuntimeException: Unable to start activity ComponentInfo{htl.at.coolcalc/htl.at
  .coolcalc.CalcActivity}: java.lang.IllegalStateException: You need to use a Theme.AppCompat theme (or descendant) with this activity.
                                                                   at android.app.ActivityThread.performLaunchActivity(ActivityThread.java:2416)
                                                                   at android.app.ActivityThread.handleLaunchActivity(ActivityThread.java:2476)
                                                                   at android.app.ActivityThread.-wrap11(ActivityThread.java)
                                                                   at android.app.ActivityThread$H.handleMessage(ActivityThread.java:1344)
                                                                   at android.os.Handler.dispatchMessage(Handler.java:102)
                                                                   at android.os.Looper.loop(Looper.java:148)
```

You need to use a Theme.AppCompat theme (or descendant) with this activity

# AppCompat entfernen

```java
package htl.at.coolcalc;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class CalcActivity extends AppCompatActivity {

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

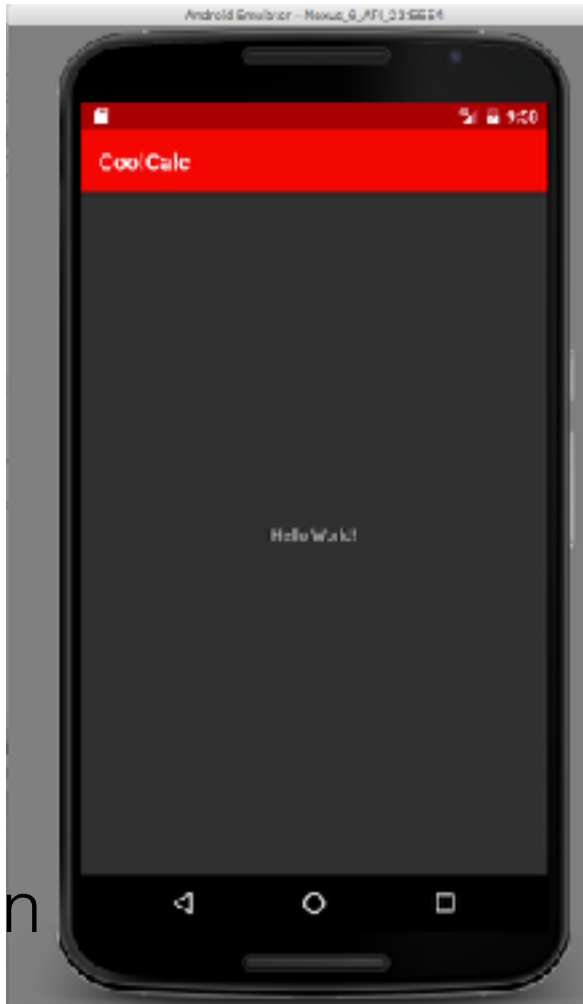
```java
package htl.at.coolcalc;

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

public class CalcActivity extends Activity {

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



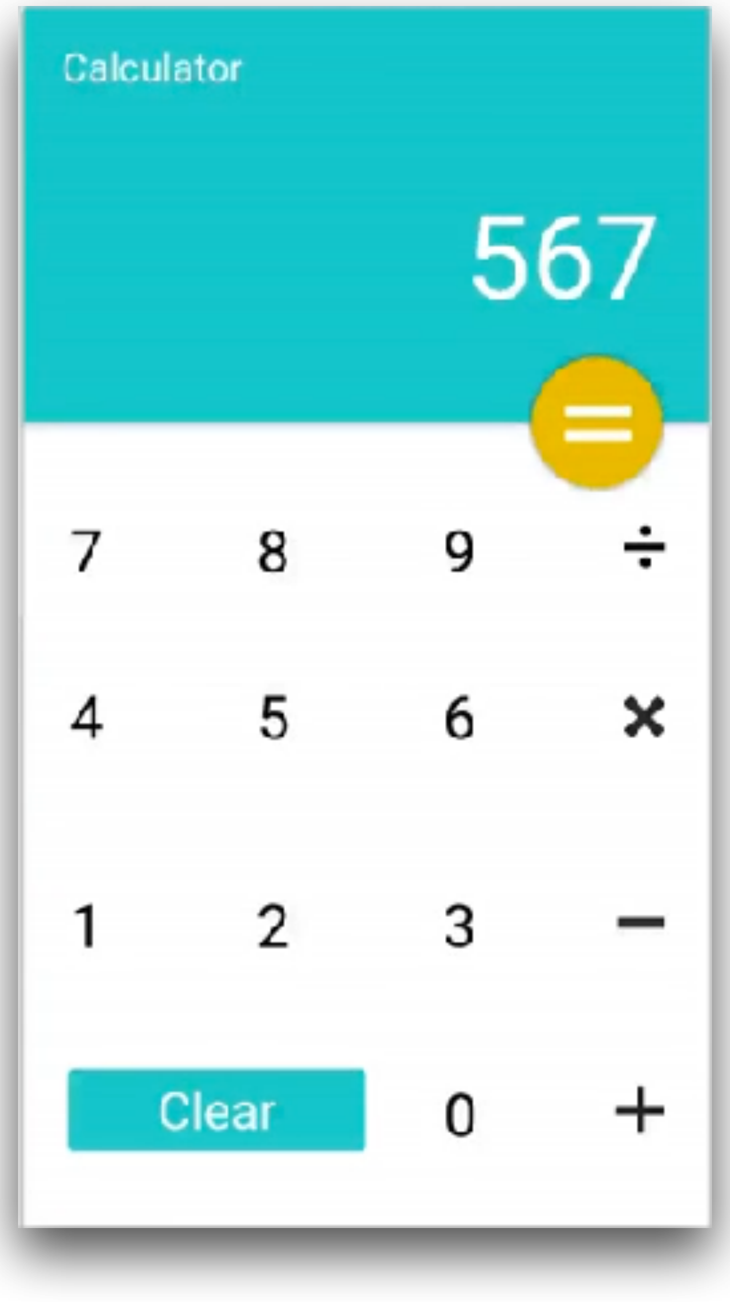
Problem: Unsere gewählten Farben wurden nicht verwendet

```xml
<resources>

    <!-- Base application theme. -->
    <style name="AppTheme" parent="@android:style/Theme.Material">
        <!-- Customize your theme here. -->
        <item name="android:colorPrimary">@color/colorPrimary</item>
        <item name="android:colorPrimaryDark">@color/colorPrimaryDark</item>
        <item name="android:colorAccent">@color/colorAccent</item>
    </style>

</resources>
```

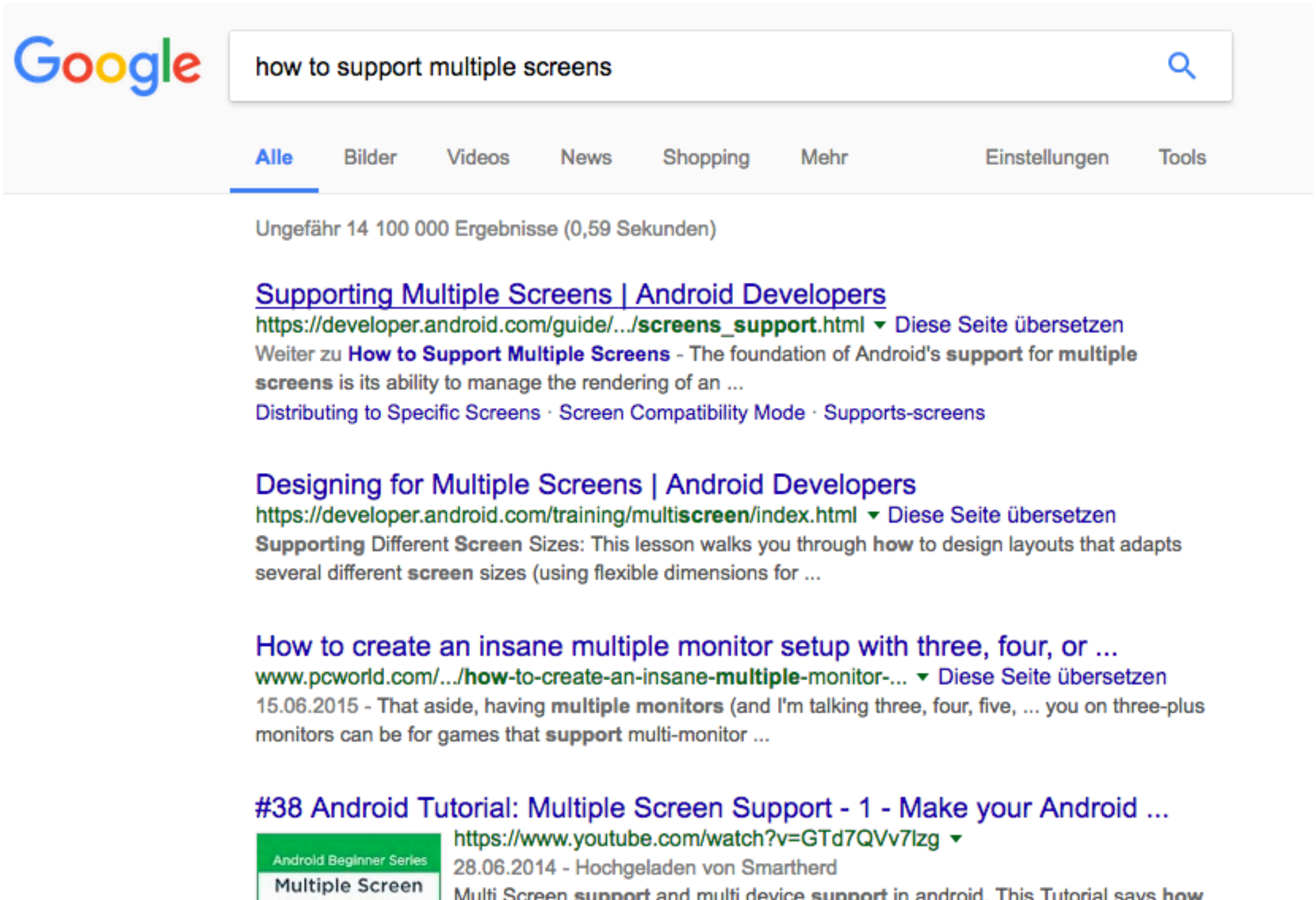
Ohne Prefix android: gelten die Farben nur für die AppCompat. Funktioniert zwar jetzt, ist aber häßlich, daher Farben auskommentieren



# Was wollen wir erstellen?



# Screen Size



how to support multiple screens

Ungefähr 14 100 000 Ergebnisse (0,59 Sekunden)

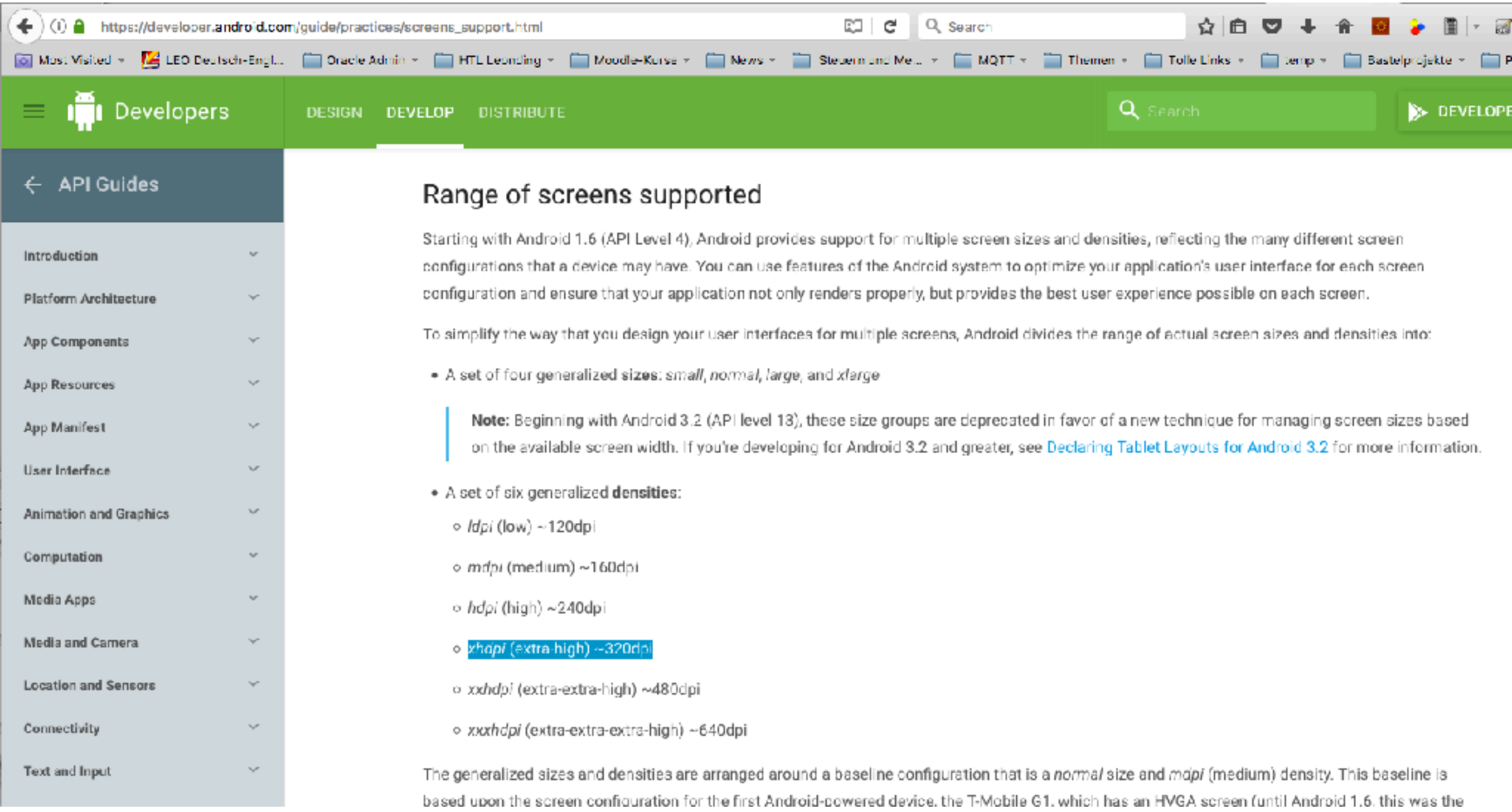
Supporting Multiple Screens | Android Developers
https://developer.android.com/guide/.../**screens_support**.html ▾ Diese Seite übersetzen
Weiter zu **How to Support Multiple Screens** - The foundation of Android's **support** for **multiple screens** is its ability to manage the rendering of an ...
Distributing to Specific Screens · Screen Compatibility Mode · Supports-screens

Designing for Multiple Screens | Android Developers
https://developer.android.com/training/multi**screen**/index.html ▾ Diese Seite übersetzen
**Supporting** Different **Screen** Sizes: This lesson walks you through **how** to design layouts that adapts several different **screen** sizes (using flexible dimensions for ...

How to create an insane multiple monitor setup with three, four, or ...
www.pcworld.com/.../**how**-to-create-an-insane-**multiple**-monitor-... ▾ Diese Seite übersetzen
15.06.2015 - That aside, having **multiple monitors** (and I'm talking three, four, five, ... you on three-plus monitors can be for games that **support** multi-monitor ...

#38 Android Tutorial: Multiple Screen Support - 1 - Make your Android ...
https://www.youtube.com/watch?v=GTd7QVv7lzg ▾
28.06.2014 - Hochgeladen von Smartherd
Multi Screen **support** and multi device **support** in android. This Tutorial says **how**

# hdpi ist „Normalgröße“



## Range of screens supported

Starting with Android 1.6 (API Level 4), Android provides support for multiple screen sizes and densities, reflecting the many different screen configurations that a device may have. You can use features of the Android system to optimize your application's user interface for each screen configuration and ensure that your application not only renders properly, but provides the best user experience possible on each screen.

To simplify the way that you design your user interfaces for multiple screens, Android divides the range of actual screen sizes and densities into:

- A set of four generalized **sizes**: *small*, *normal*, *large*, and *xlarge*

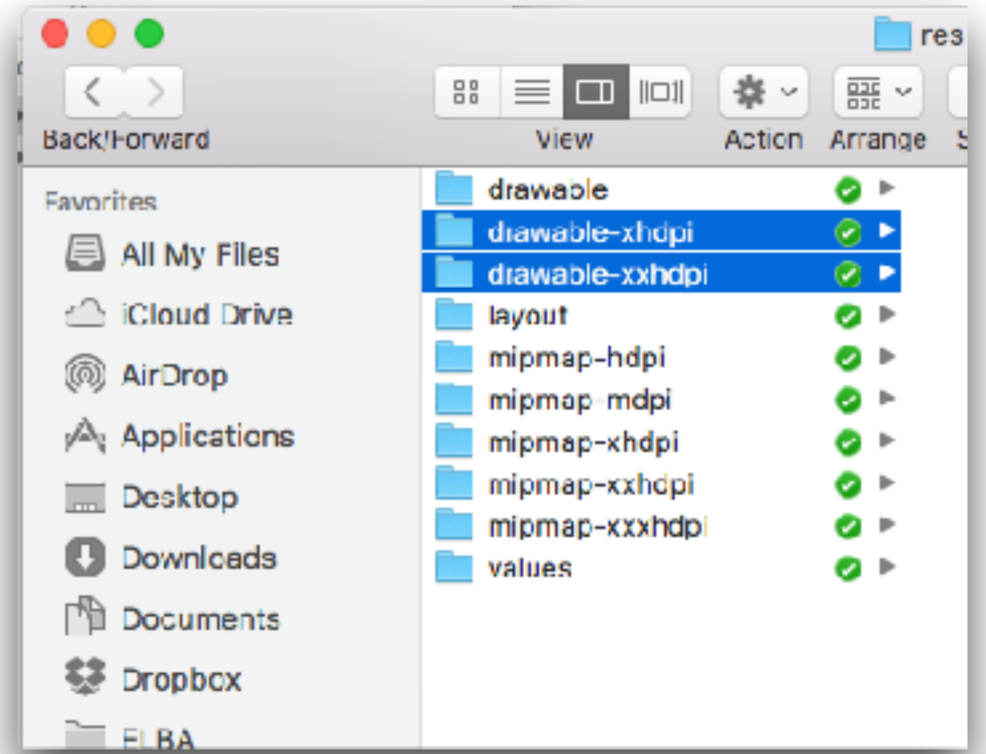
  > **Note:** Beginning with Android 3.2 (API level 13), these size groups are deprecated in favor of a new technique for managing screen sizes based on the available screen width. If you're developing for Android 3.2 and greater, see Declaring Tablet Layouts for Android 3.2 for more information.

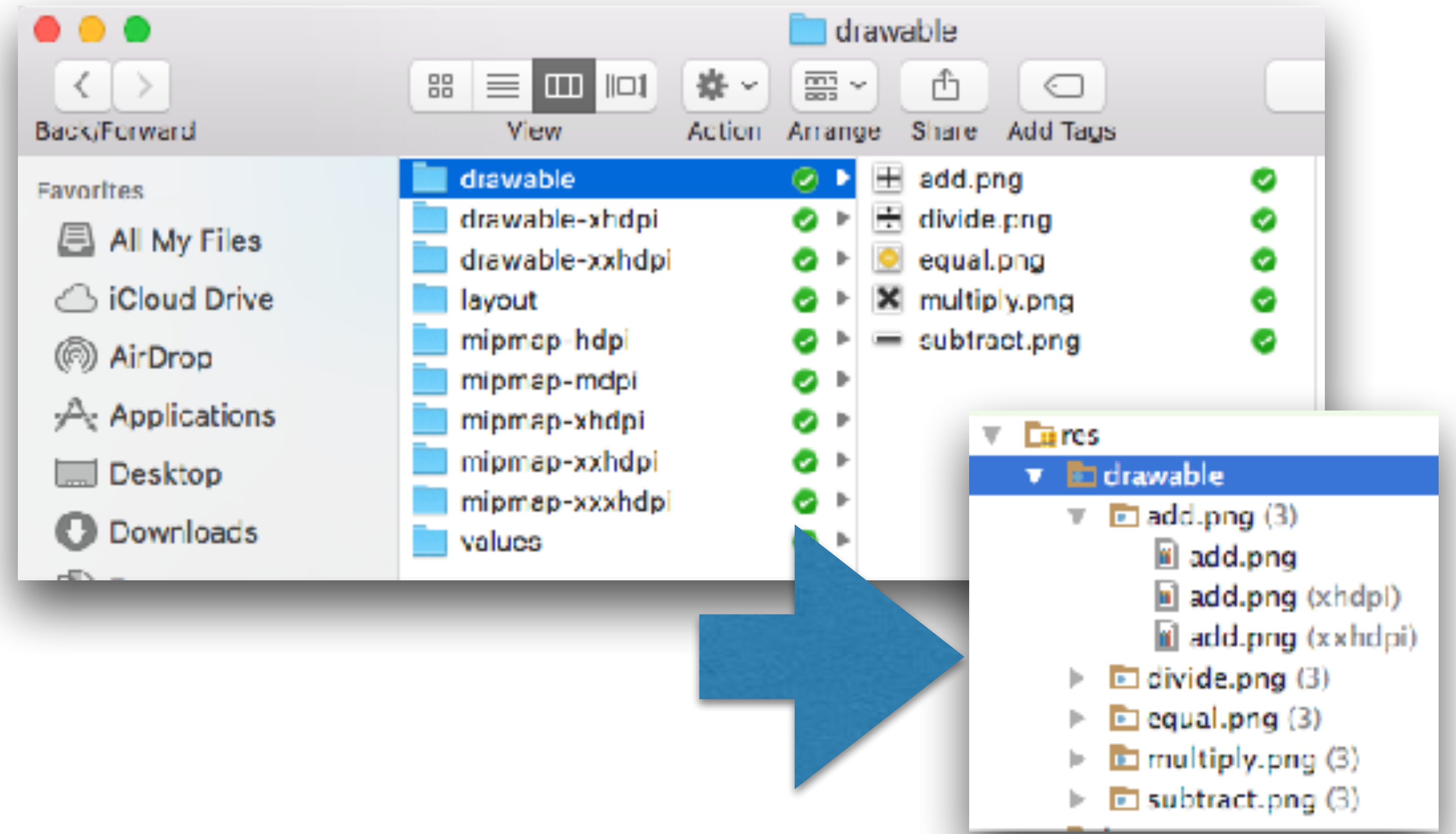
- A set of six generalized **densities**:
  - *ldpi* (low) ~120dpi
  - *mdpi* (medium) ~160dpi
  - *hdpi* (high) ~240dpi
  - *xhdpi* (extra-high) ~320dpi
  - *xxhdpi* (extra-extra-high) ~480dpi
  - *xxxhdpi* (extra-extra-extra-high) ~640dpi

The generalized sizes and densities are arranged around a baseline configuration that is a *normal* size and *mdpi* (medium) density. This baseline is based upon the screen configuration for the first Android-powered device, the T-Mobile G1, which has an HVGA screen (until Android 1.6, this was the

# Neue Folder anlegen

- Rechtsklick auf res/drawable - Reveal in finder
- Anlegen zweier neu Folder: drawable-xhdpi und drawable - xxhdpi
- Kopieren der Files in die Ordner (1x, 2x, 3x)

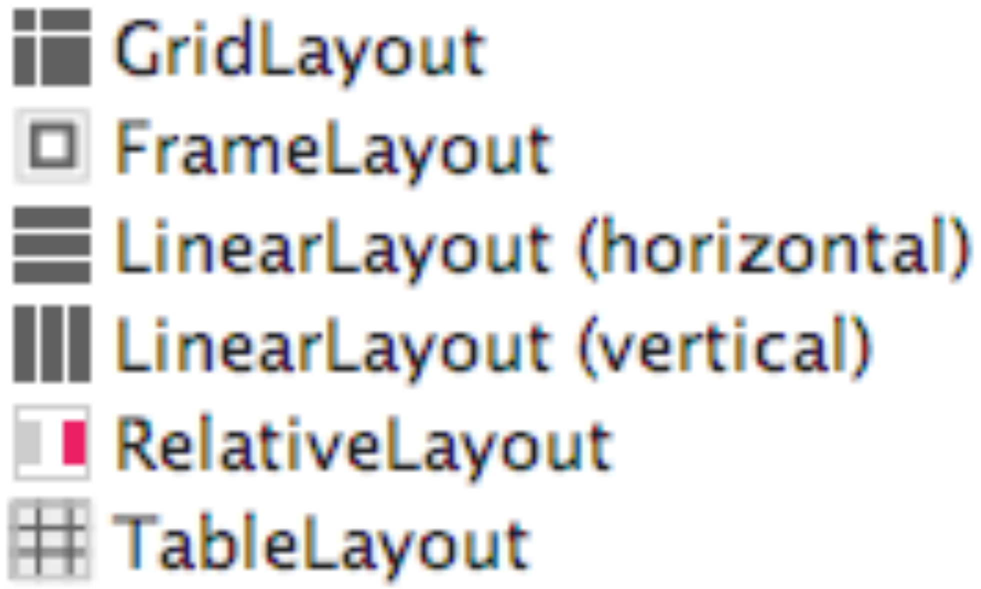


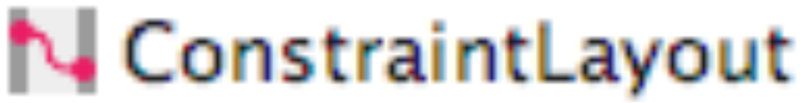


# Layouts

alt

neu


GridLayout
FrameLayout
LinearLayout (horizontal)
LinearLayout (vertical)
RelativeLayout
TableLayout


ConstraintLayout

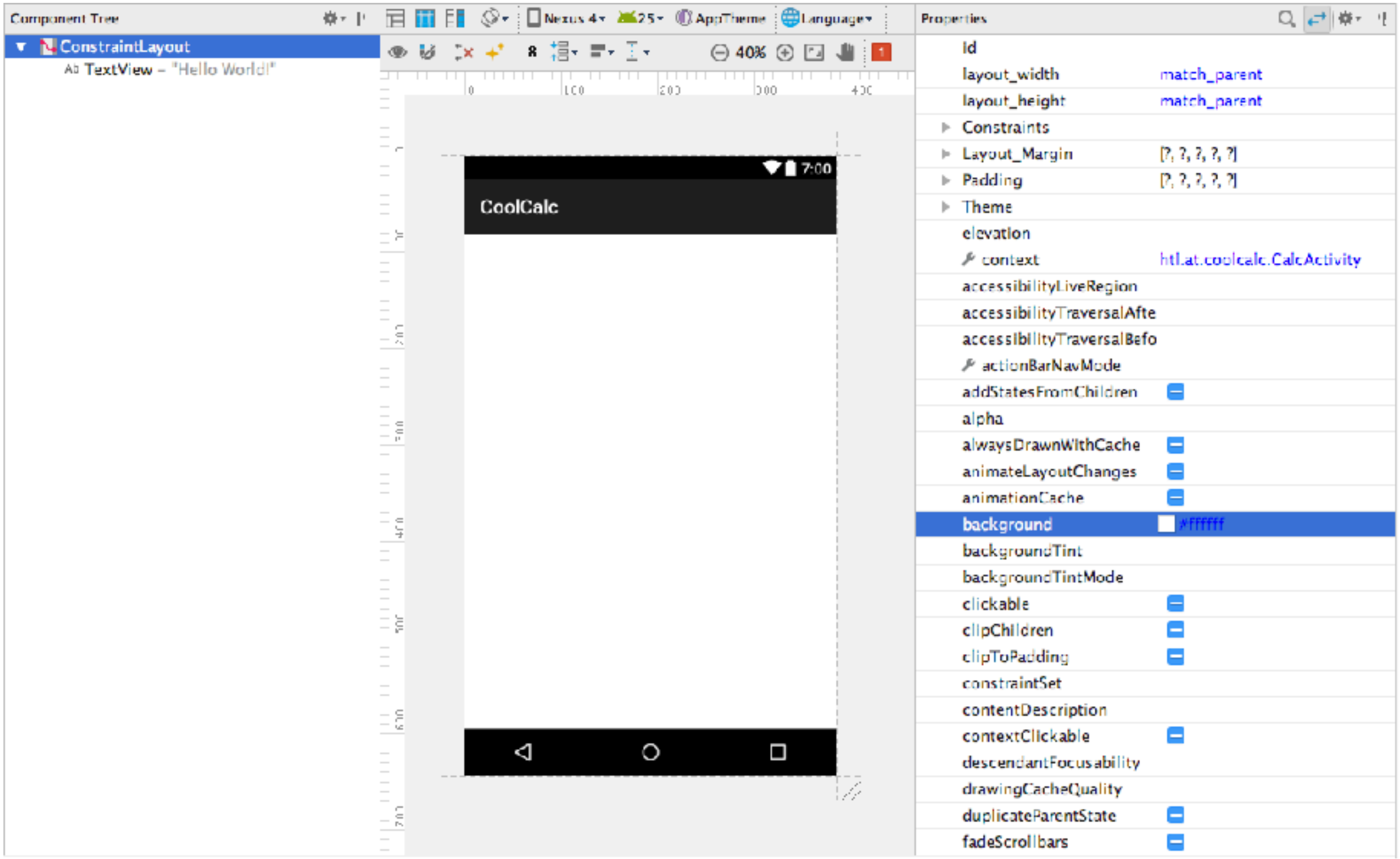
https://riggaroo.co.za/constraintlayout-101-new-layout-builder-android-studio/

https://medium.com/exploring-android/exploring-the-new-android-constraintlayout-eed37fe8d8f1#.k3zc6fe1l

https://medium.com/@loutry/guide-to-constraintlayout-407cd87bc013#.y9lqz7ie5

http://wiresareobsolete.com/2016/07/constraintlayout-part-1/

# Hintergrundfarbe



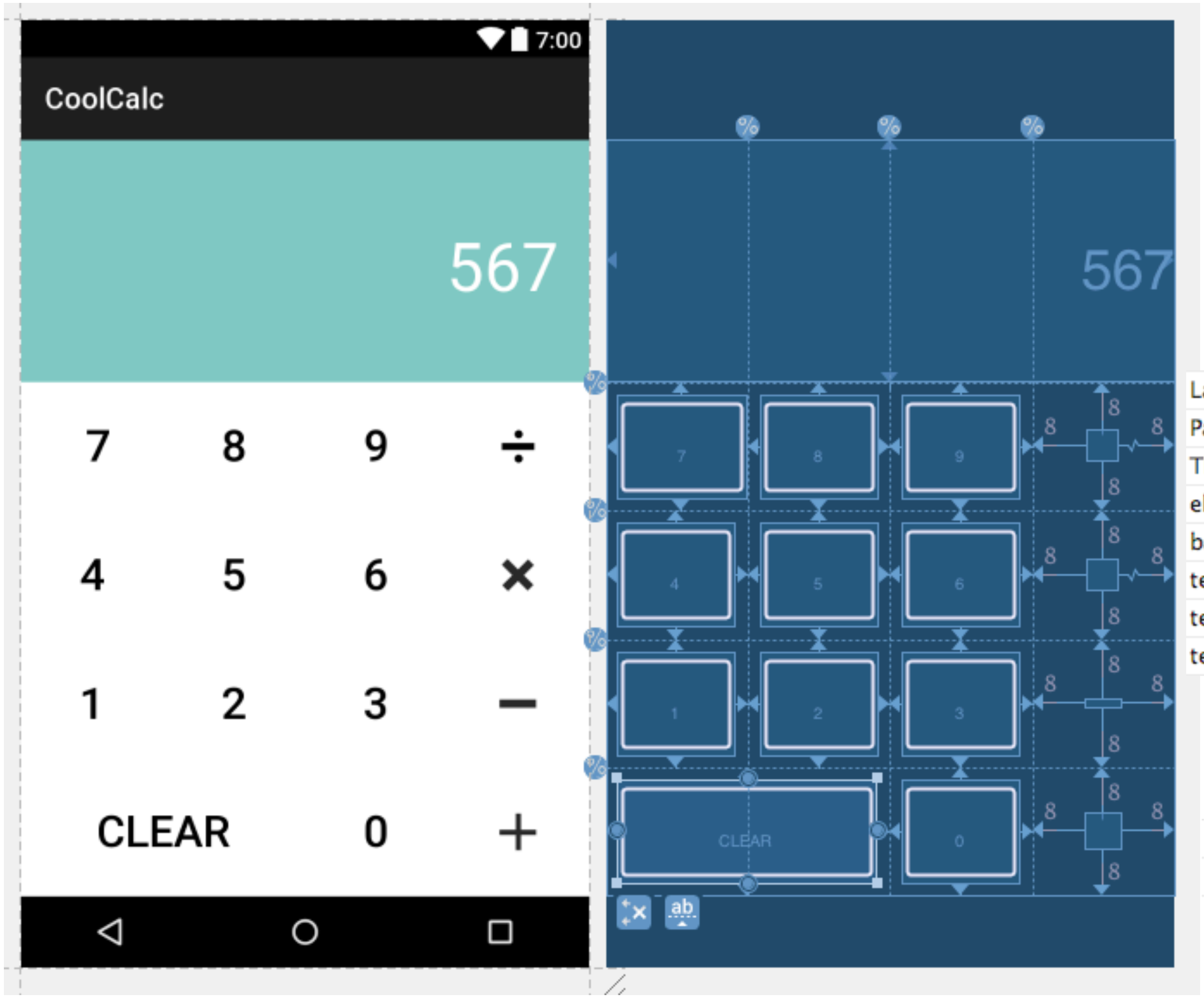
Component Tree

- ConstraintLayout
  - TextView – "Hello World!"

Properties

| | |
|---|---|
| id | |
| layout_width | match_parent |
| layout_height | match_parent |
| Constraints | |
| Layout_Margin | [?, ?, ?, ?, ?] |
| Padding | [?, ?, ?, ?, ?] |
| Theme | |
| elevation | |
| context | htl.at.coolcalc.CalcActivity |
| accessibilityLiveRegion | |
| accessibilityTraversalAfte | |
| accessibilityTraversalBefo | |
| actionBarNavMode | |
| addStatesFromChildren | ☐ |
| alpha | |
| alwaysDrawnWithCache | ☐ |
| animateLayoutChanges | ☐ |
| animationCache | ☐ |
| background | ☐ #ffffff |
| backgroundTint | |
| backgroundTintMode | |
| clickable | ☐ |
| clipChildren | ☐ |
| clipToPadding | ☐ |
| constraintSet | |
| contentDescription | |
| contextClickable | ☐ |
| descendantFocusability | |
| drawingCacheQuality | |
| duplicateParentState | ☐ |
| fadeScrollbars | ☐ |

# Layout erstellen



## Button

| | |
|---|---|
| background | ⬜ #ffffff |
| text | 7 |
| textColor | ⬛ #000000 |
| textSize | 30sp |

## TextView

| | |
|---|---|
| Layout_Margin | [?, 32dp, 16dp, 0dp, 16dp |
| Padding | [?, ?, ?, ?, ?] |
| Theme | |
| elevation | |
| background | 🟦 #38c5c6 |
| text | Clear |
| textColor | ⬜ #ffffff |
| textSize | 25sp |

## ImageButton

| | |
|---|---|
| background | ⬜ #00ffffff |
| srcCompat | @drawable/divide |

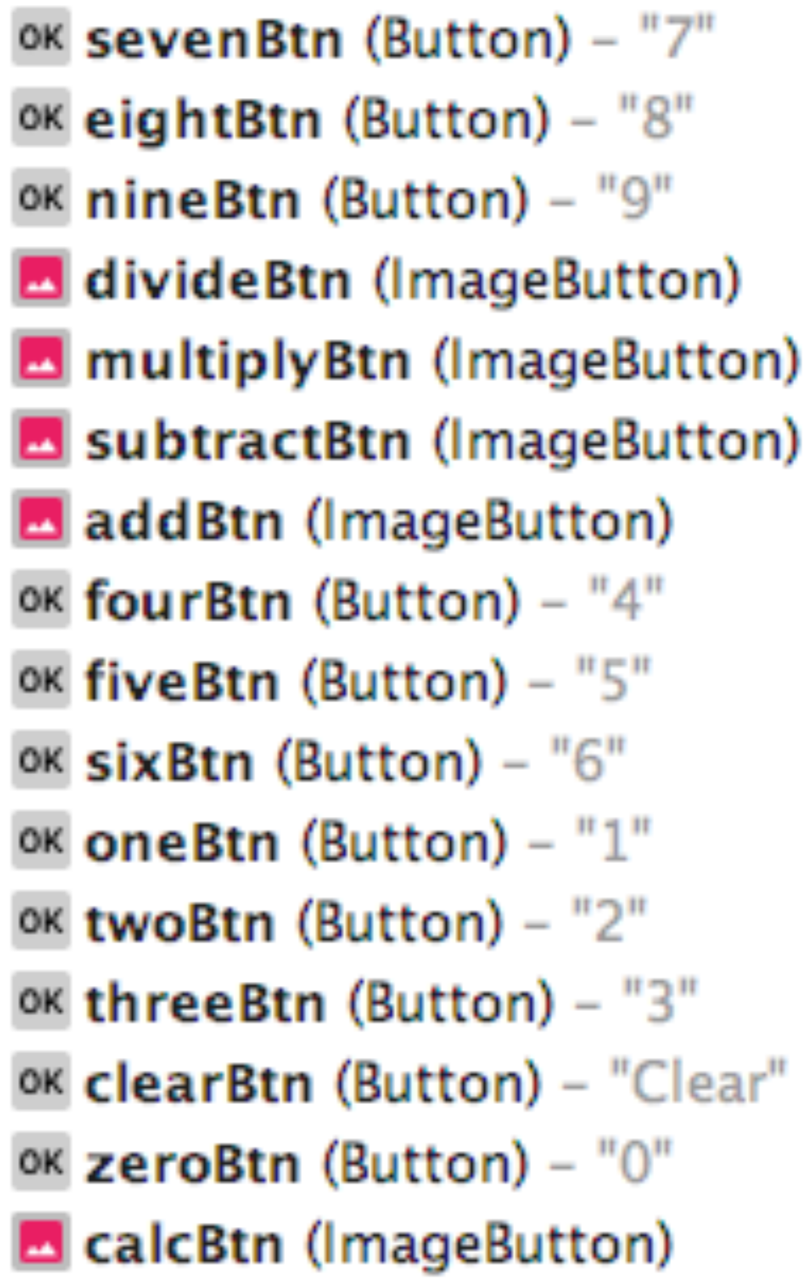
# Problem: ImageButtons nicht sichtbar

```
<ImageButton
    android:id="@+id/imageButton5"
    android:layout_width="80dp"
    android:layout_height="80dp"
    android:layout_marginRight="16dp"
    android:layout_marginTop="8dp"
    android:background="#00ffffff"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:srcCompat="@drawable/equal"
    android:layout_marginBottom="8dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintVertical_bias="0.243" />
```

```
<ImageButton
    android:id="@+id/imageButton5"
    android:layout_width="80dp"
    android:layout_height="80dp"
    android:layout_marginRight="16dp"
    android:layout_marginTop="8dp"
    android:background="#00ffffff"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    android:src="@drawable/equal"
    android:layout_marginBottom="8dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintVertical_bias="0.243" />
```

„app:srcCompat“ zu „android:src“ ändern

# Benennen der Steuerelemente



```
OK sevenBtn (Button) – "7"
OK eightBtn (Button) – "8"
OK nineBtn (Button) – "9"
   divideBtn (ImageButton)
   multiplyBtn (ImageButton)
   subtractBtn (ImageButton)
   addBtn (ImageButton)
OK fourBtn (Button) – "4"
OK fiveBtn (Button) – "5"
OK sixBtn (Button) – "6"
OK oneBtn (Button) – "1"
OK twoBtn (Button) – "2"
OK threeBtn (Button) – "3"
OK clearBtn (Button) – "Clear"
OK zeroBtn (Button) – "0"
   calcBtn (ImageButton)
```

# Steuerelemente im Code bekannt machen

```java
public class CalcActivity extends Activity {

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

        Button oneBtn = (Button) findViewById(R.id.oneBtn);
        Button twoBtn = (Button) findViewById(R.id.twoBtn);
        Button threeBtn = (Button) findViewById(R.id.threeBtn);
        Button fourBtn = (Button) findViewById(R.id.fourBtn);
        Button fiveBtn = (Button) findViewById(R.id.fiveBtn);
        Button sixBtn = (Button) findViewById(R.id.sixBtn);
        Button sevenBtn = (Button) findViewById(R.id.sevenBtn);
        Button eightBtn = (Button) findViewById(R.id.eightBtn);
        Button nineBtn = (Button) findViewById(R.id.nineBtn);
        Button zeroBtn = (Button) findViewById(R.id.zeroBtn);

        ImageButton calcBtn = (ImageButton) findViewById(R.id.calcBtn);
        ImageButton divideBtn = (ImageButton) findViewById(R.id.divideBtn);
        ImageButton multiplyBtn = (ImageButton) findViewById(R.id.multiplyBtn);
        ImageButton subtractBtn = (ImageButton) findViewById(R.id.subtractBtn);
        ImageButton addBtn = (ImageButton) findViewById(R.id.addBtn);
    }
}
```

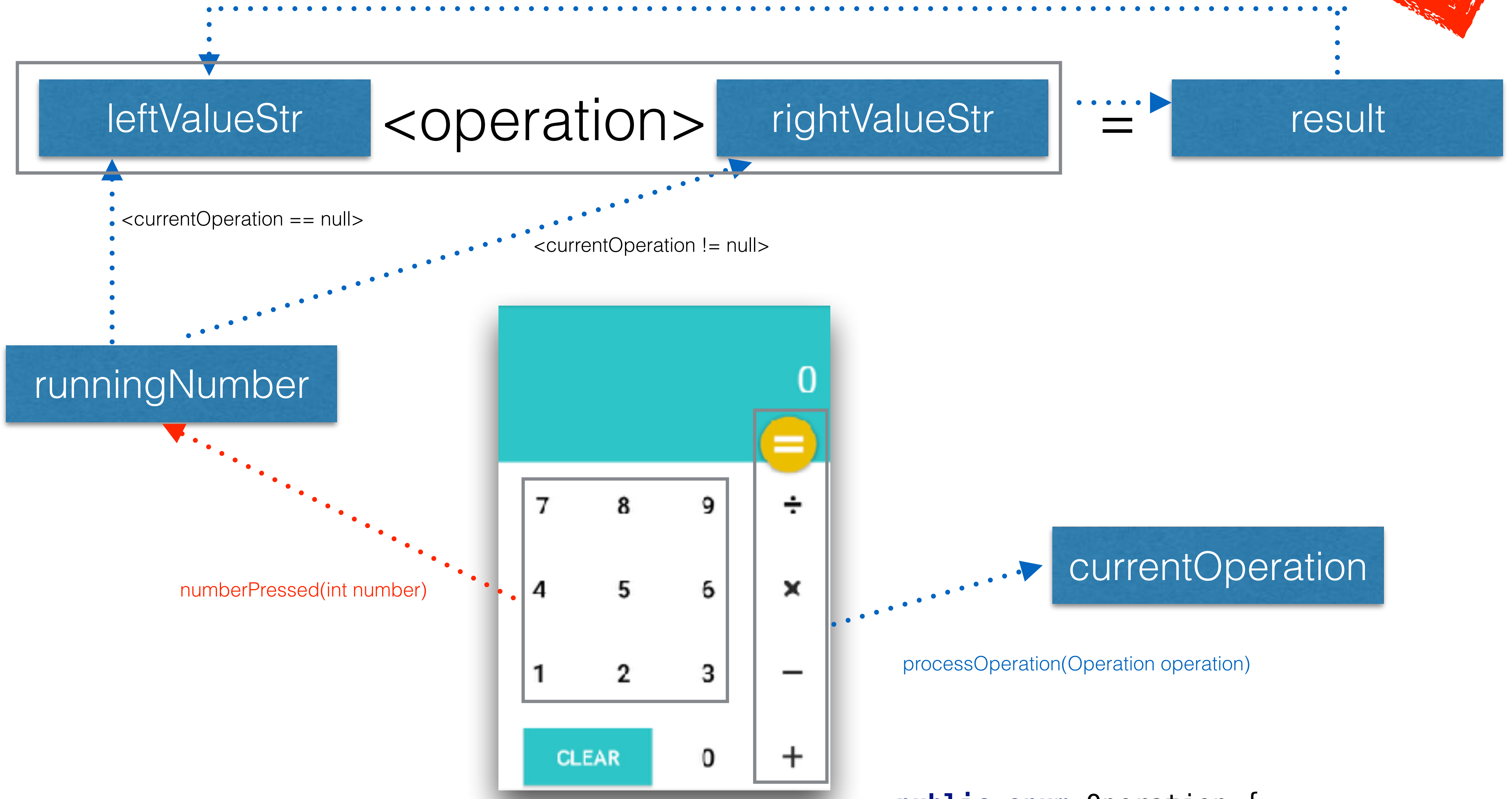
Alternativen:
Butterknive
androidannotations
(toothpick)
oder
**Kotlin**

# OnClickListener implementieren

```
oneBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

    }
});

twoBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

    }
});

threeBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

    }
});

...
```

# Algorithmus

Aufgabe



leftValueStr
<operation>
rightValueStr
=
result
<currentOperation == null>
<currentOperation != null>
runningNumber
0
7 8 9
4 5 6
1 2 3
CLEAR 0
÷ × − +
numberPressed(int number)
currentOperation
processOperation(Operation operation)

```
public enum Operation {
    ADD, SUBTRACT, DIVIDE, MULTIPLY, EQUAL
}
```

## Noch Fragen?