WeatherViewer App

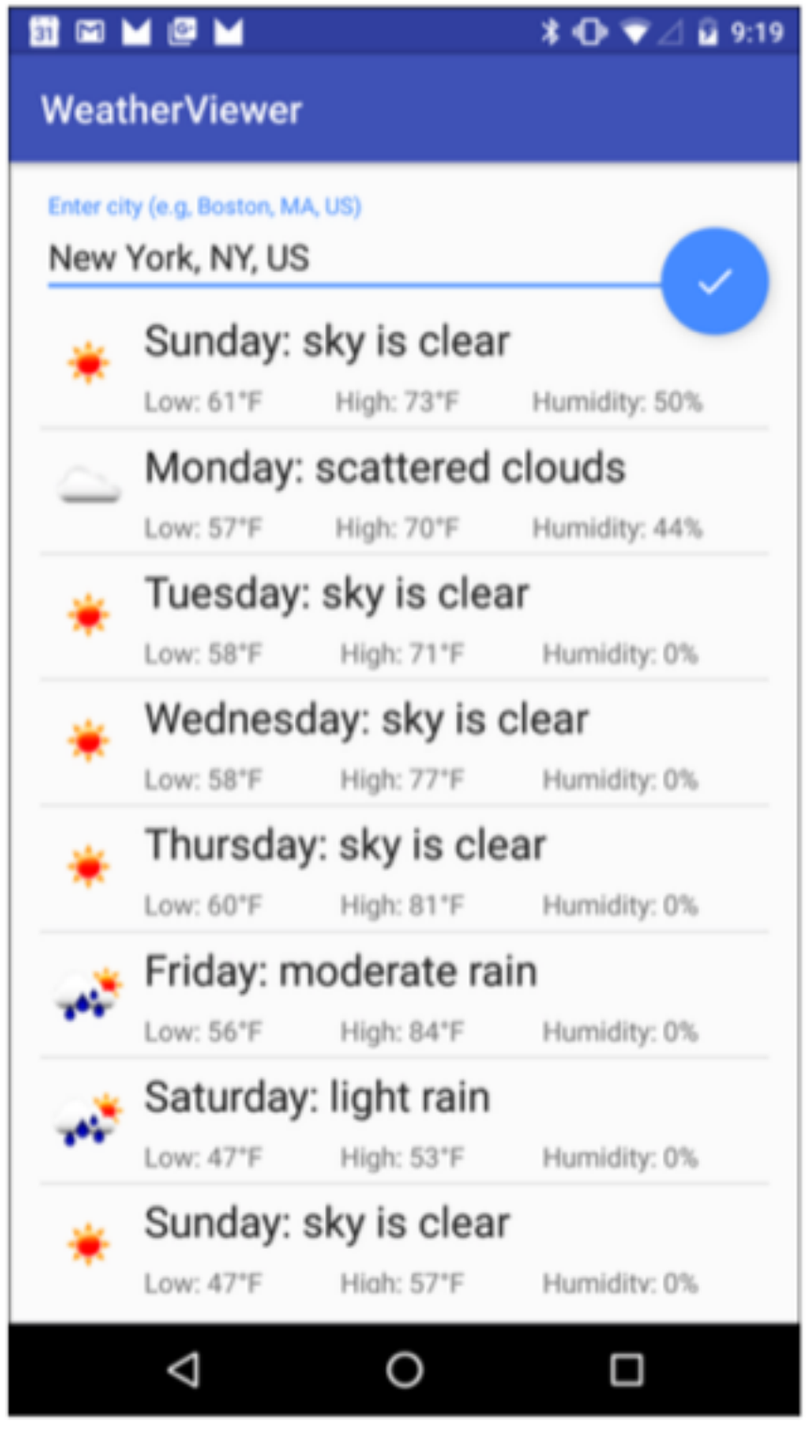
REST Web Services, AsyncTask, HttpUrlConnection, Processing JSON Responses, JSONObject, JSONArray, ListView, ArrayAdapter, ViewHolder Pattern, TextInputLayout, FloatingActionButton

Übersicht

- Inhalte der App
  - Wetterabhängiges Icon
  - Wettervorhersage
  - Höchste und niedrigste Tagestemperaturen
  - Feuchtigkeit in %

http://openweathermap.org/forecast16

http://openweathermap.org/api



OpenWeatherMap.org

- OpenWeatherMap.org uses a creative commons public license for its web services. For the license terms, visit:
  http://creativecommons.org/licenses/by-sa/2.0/

- For more information about the license terms, see the Licenses section at
  http://openweathermap.org/terms

- Obtaining an OpenWeatherMap.org API Key
  Before running this app, you must obtain your own OpenWeatherMap.org API key from
  http://openweathermap.org/register

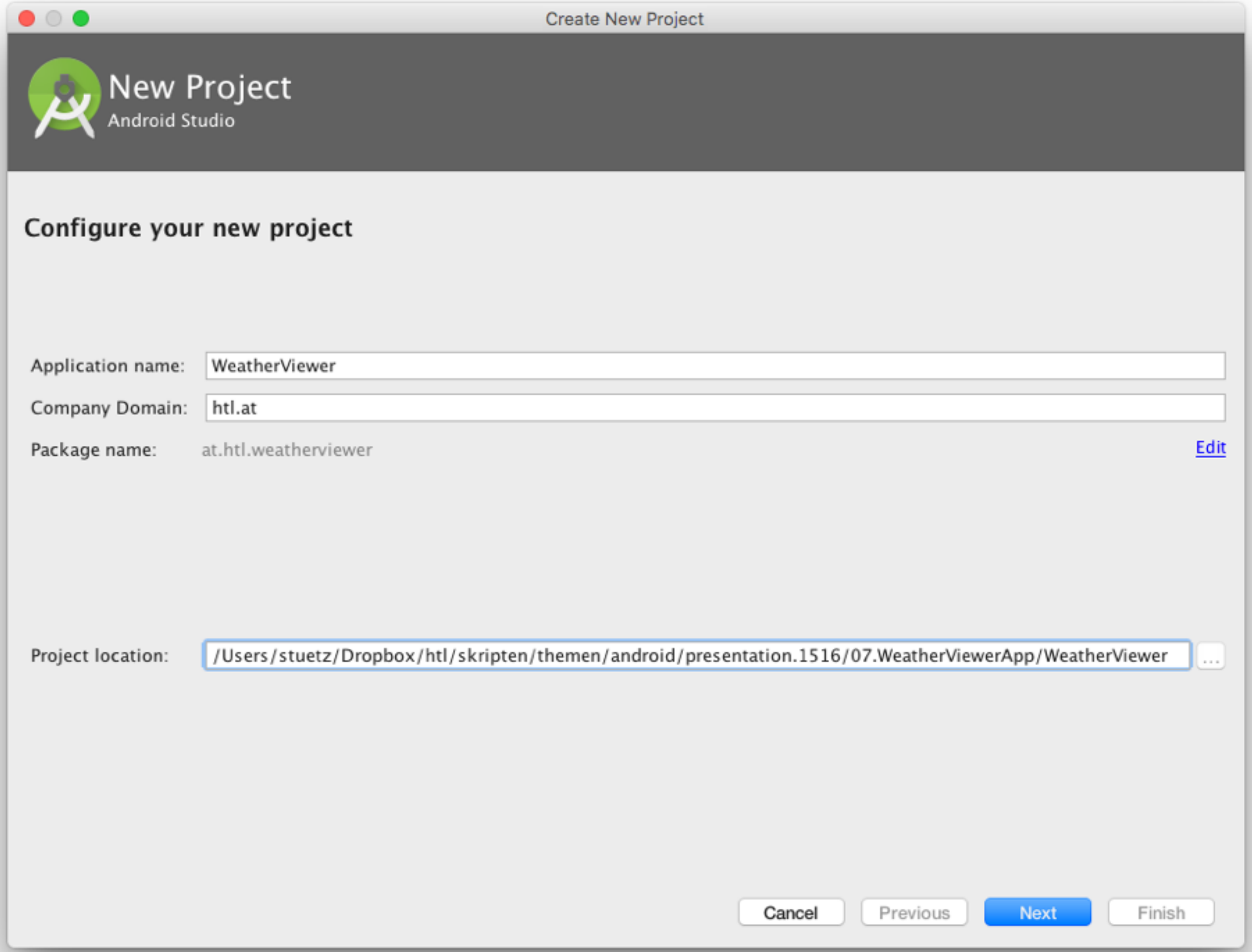
- After registering, copy the hexadecimal API key from the confirmation web page, then re- place YOUR_API_KEY in strings.xml with the key.

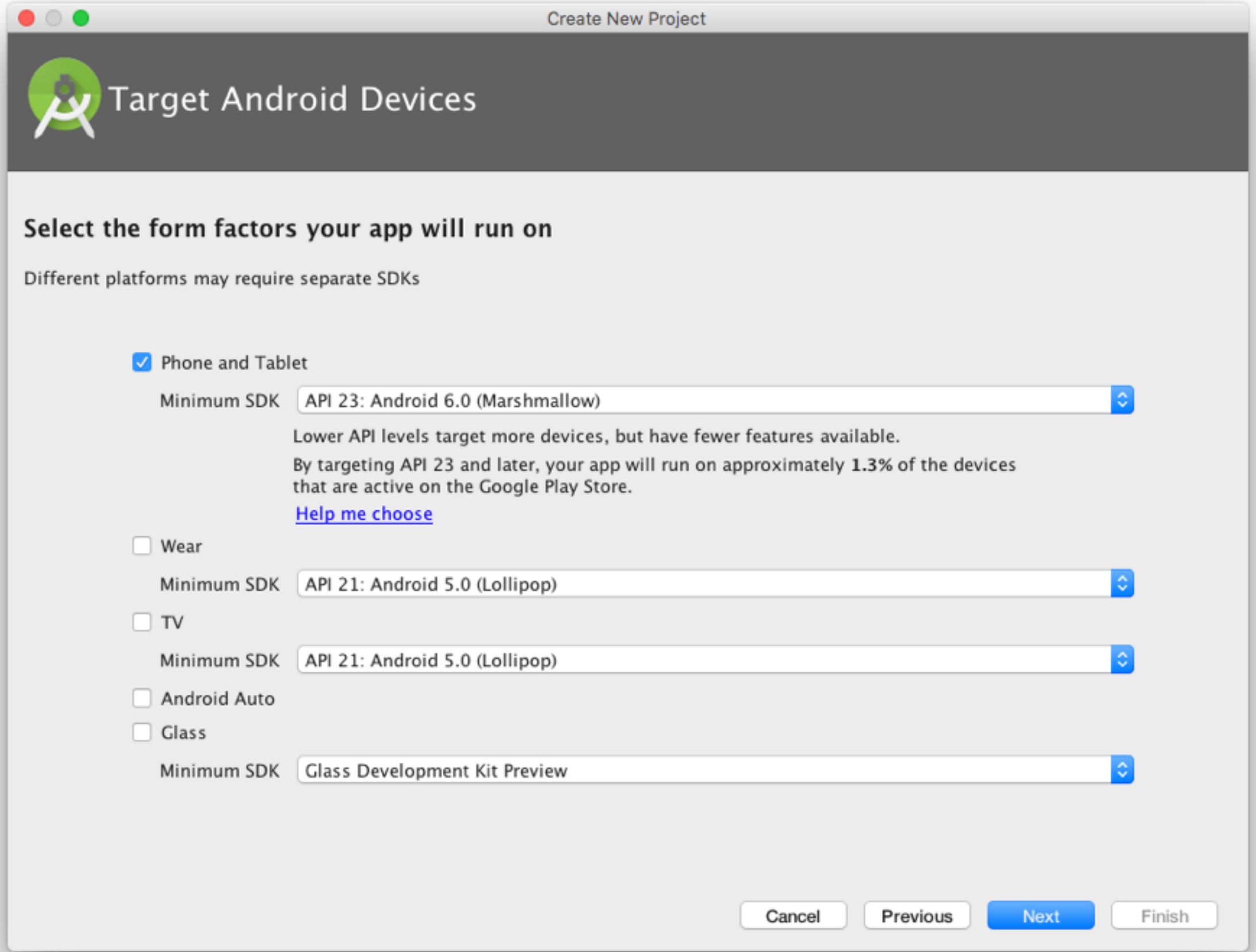
JSON


```
{
  "city":{
    "id":7872055,
    "name":"Leonding",
    "coord":{
      "lon":14.25467,
      "lat":48.279942
    },
    "country":"AT",
    "population":0
  },
  "cod":"200",
  "message":0.0746,
  "cnt":16,
  "list":[
  {
    "dt":1457953200,
    "temp":{
      "day":5.21,
      "min":-3.86,
      "max":5.21,
      "night":-3.86,
      "eve":1.74,
      "morn":5.21
    },
    "pressure":972.97,
    "humidity":87,
    "weather":[
      {
        "id":800,
        "main":"Clear",
        "description":"clear sky",
        "icon":"01d"
      }
    ],
    "speed":3.12,
    "deg":82,
    "clouds":0
  },
```

Technologie Übersicht

- WebServices (RESTful)
- JSON
- HttpUrlConnection
- AsyncTask
- ListView, ArrayAdapter, View-Holder Pattern
- FloatingActionButton
- Snackbar (InteraktiverToast)


Create New Project
New Project
Android Studio
Configure your new project
Application name:
WeatherViewer
Company Domain:
htl.at
Package name:
at.htl.weatherviewer
Edit
Project location:
/Users/stuetz/Dropbox/htl/skripten/themen/android/presentation.1516/07.WeatherViewerApp/WeatherViewer
...
Cancel
Previous
Next
Finish


Create New Project
Target Android Devices
Select the form factors your app will run on
Different platforms may require separate SDKs
Phone and Tablet
Minimum SDK
API 23: Android 6.0 (Marshmallow)
Lower API levels target more devices, but have fewer features available.
By targeting API 23 and later, your app will run on approximately 1.3% of the devices that are active on the Google Play Store.
Help me choose
Wear
Minimum SDK
API 21: Android 5.0 (Lollipop)
TV
Minimum SDK
API 21: Android 5.0 (Lollipop)
Android Auto
Glass
Minimum SDK
Glass Development Kit Preview
Cancel
Previous
Next
Finish

Create New Project

Add an Activity to Mobile

Add No Activity

Basic Activity

Empty Activity

Fullscreen Activity

Google AdMob Ads Activity

Google Maps Activity

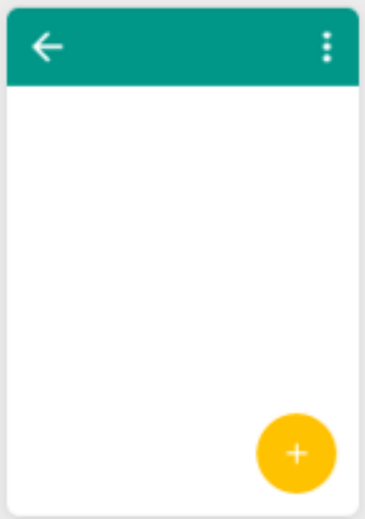
Login Activity

Master/Detail Flow

Cancel | Previous | Next | Finish

Create New Project

Customize the Activity



Basic Activity

Creates a new basic activity with an app bar.

Activity Name: MainActivity

Layout Name: activity_main

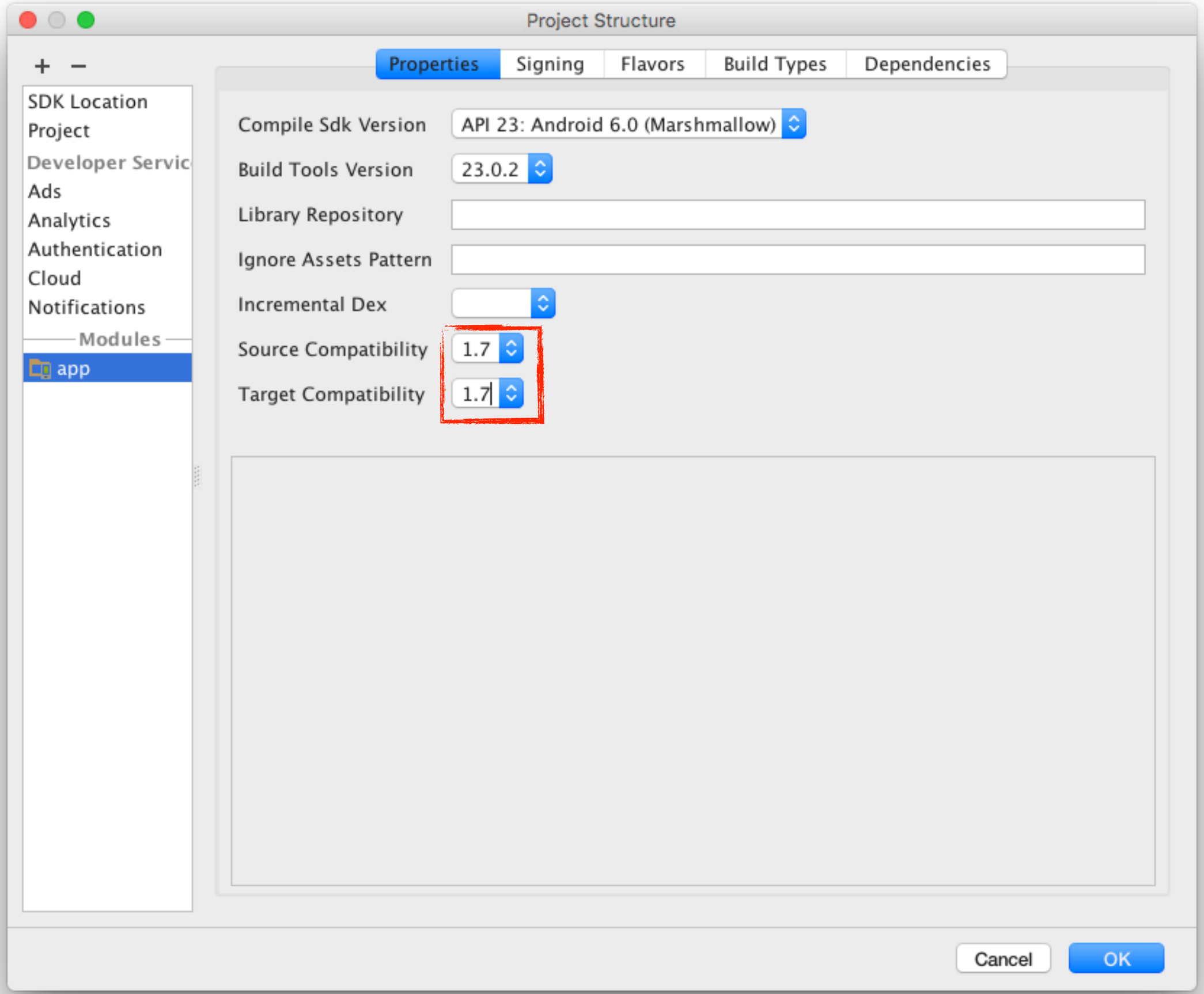
Title: MainActivity

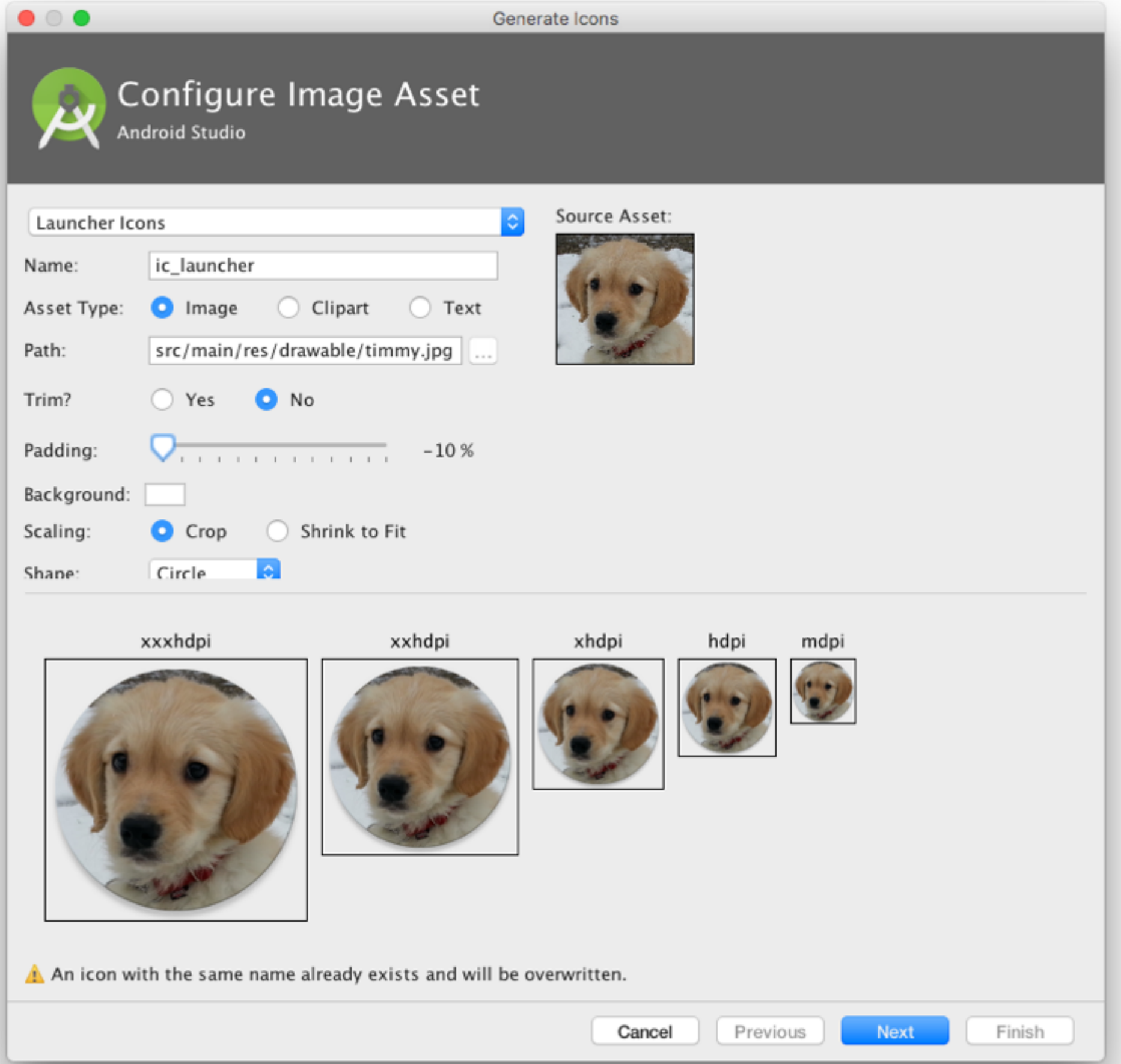
Menu Resource Name: menu_main

Use a Fragment

If true, the content will be a fragment

Cancel | Previous | Next | Finish


Project Structure
Properties
Signing
Flavors
Build Types
Dependencies
SDK Location
Project
Developer Servic
Ads
Analytics
Authentication
Cloud
Notifications
Modules
app
Compile Sdk Version
API 23: Android 6.0 (Marshmallow)
Build Tools Version
23.0.2
Library Repository
Ignore Assets Pattern
Incremental Dex
Source Compatibility
1.7
Target Compatibility
1.7
Cancel
OK


Generate Icons
Configure Image Asset
Android Studio
Launcher Icons
Name:
ic_launcher
Asset Type:
Image
Clipart
Text
Path:
src/main/res/drawable/timmy.jpg
...
Trim?
Yes
No
Padding:
-10 %
Background:
Scaling:
Crop
Shrink to Fit
Shape:
Circle
Source Asset:
xxxhdpi
xxhdpi
xhdpi
hdpi
mdpi
An icon with the same name already exists and will be overwritten.
Cancel
Previous
Next
Finish

Permissions

- Ab dem Android 6.0 Permission-Modell wird einer App automatisch der Zugriff auf das Internet gewährt, da dies mittlerweile als fundamentale Anforderung angesehen wird.

- Andere Permissions, die ebenfalls nunmehr von Haus aus gewährt werden, findet man unter

http://developer.android.com/guide/topics/security/permissions.html

AndroidManifest.xml

AndroidManifest.xml

manifest | application | activity

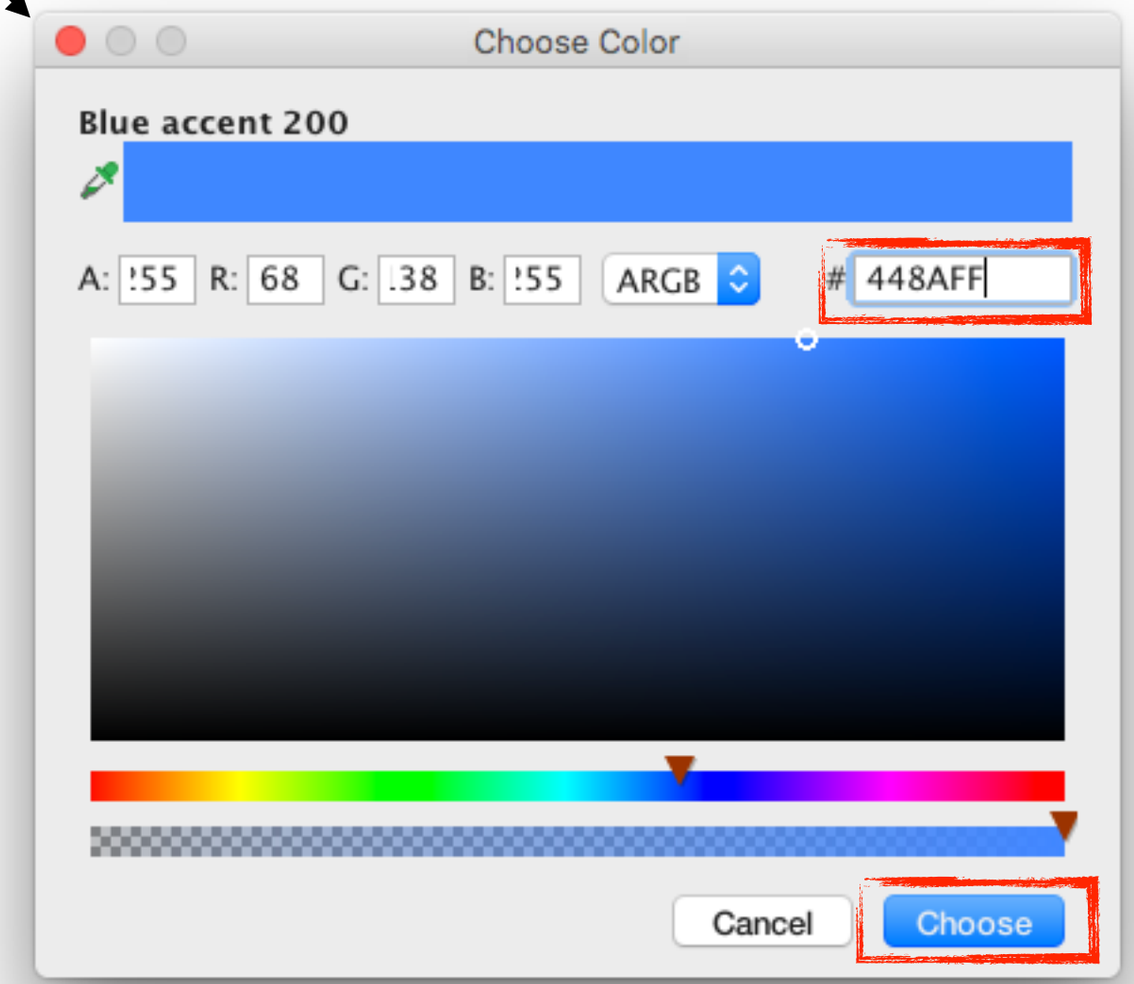
```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      package="at.htl.weatherviewer">
4
5      <uses-permission android:name="android.permission.INTERNET" />
6
7      <application
8          android:allowBackup="true"
9          android:icon="@mipmap/ic_launcher"
10         android:label="WeatherViewer"
11         android:supportsRtl="true"
12         android:theme="@style/AppTheme">
13         <activity
14             android:name=".MainActivity"
15             android:label="WeatherViewer"
16             android:screenOrientation="portrait"
17             android:theme="@style/AppTheme.NoActionBar">
18             <intent-filter>
19                 <action android:name="android.intent.action.MAIN" />
20
21                 <category android:name="android.intent.category.LAUNCHER" />
22             </intent-filter>
23         </activity>
24     </application>
25
26 </manifest>
```

Variante 1

colorAccent ändern

colors.xml

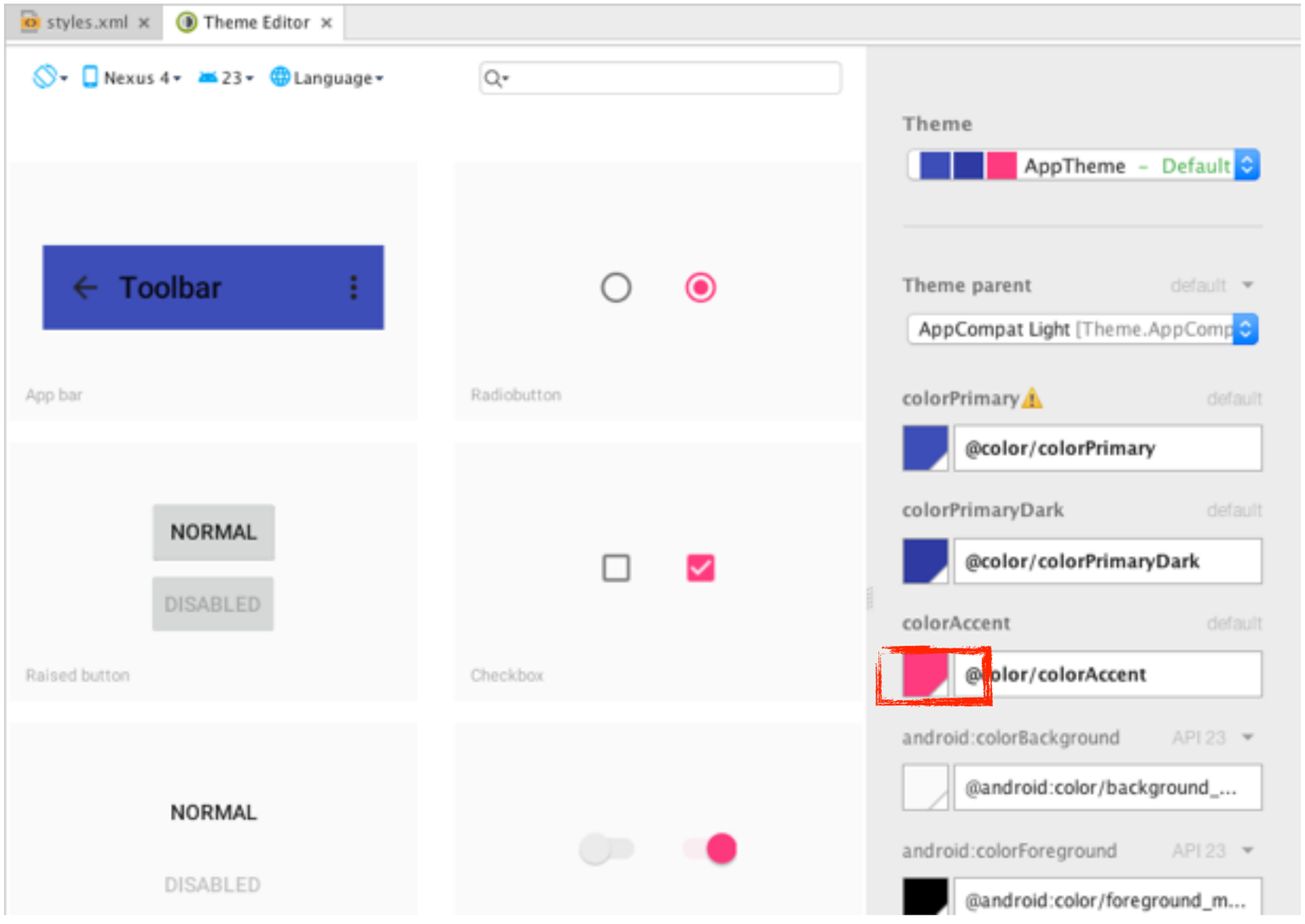
```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#3F51B5</color>
    <color name="colorPrimaryDark">#303F9F</color>
    <color name="colorAccent">#FF4081</color>
</resources>
```



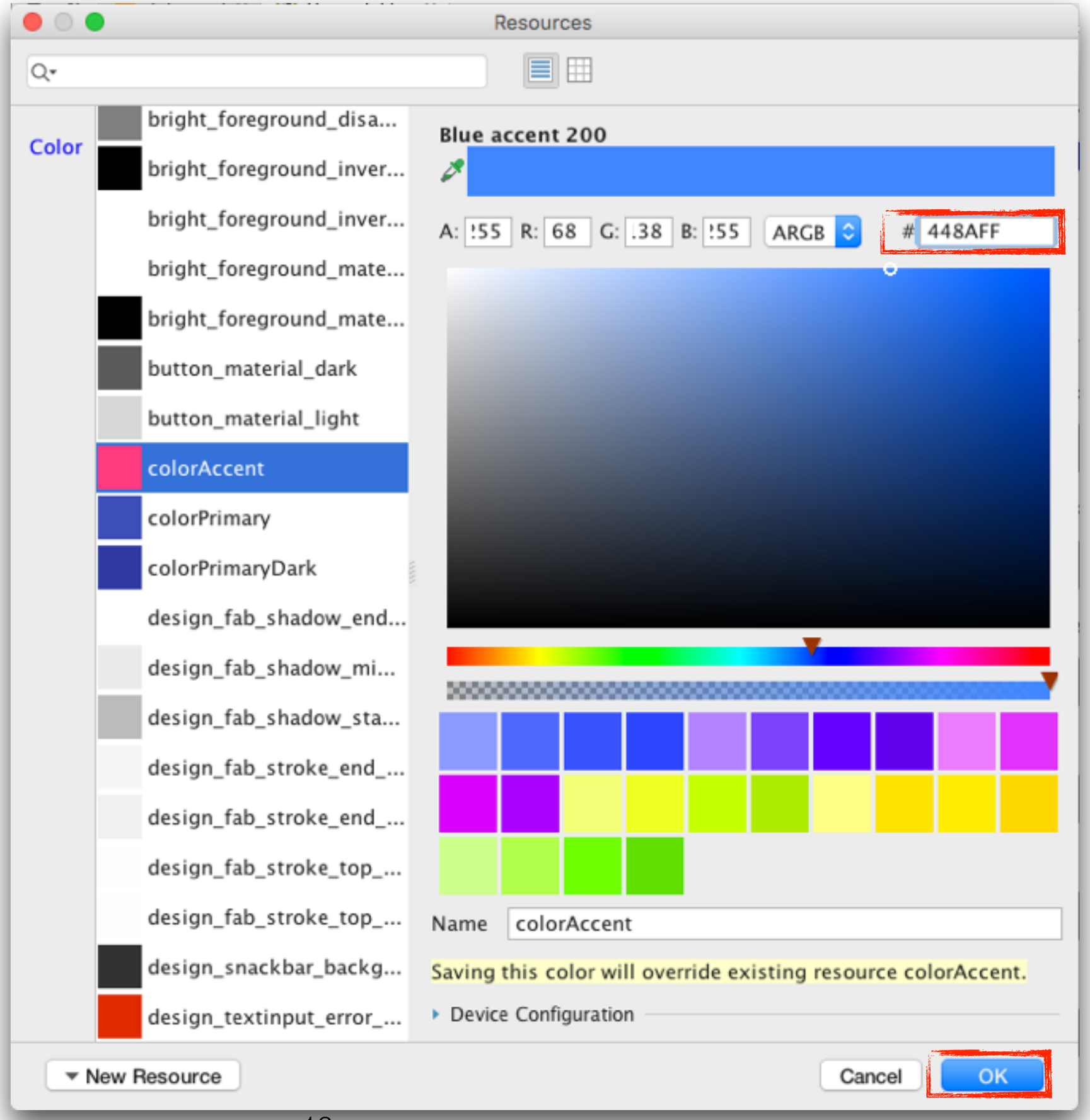

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#3F51B5</color>
    <color name="colorPrimaryDark">#303F9F</color>
    <color name="colorAccent">#448aff</color>
</resources>
```

colorAccent ändern

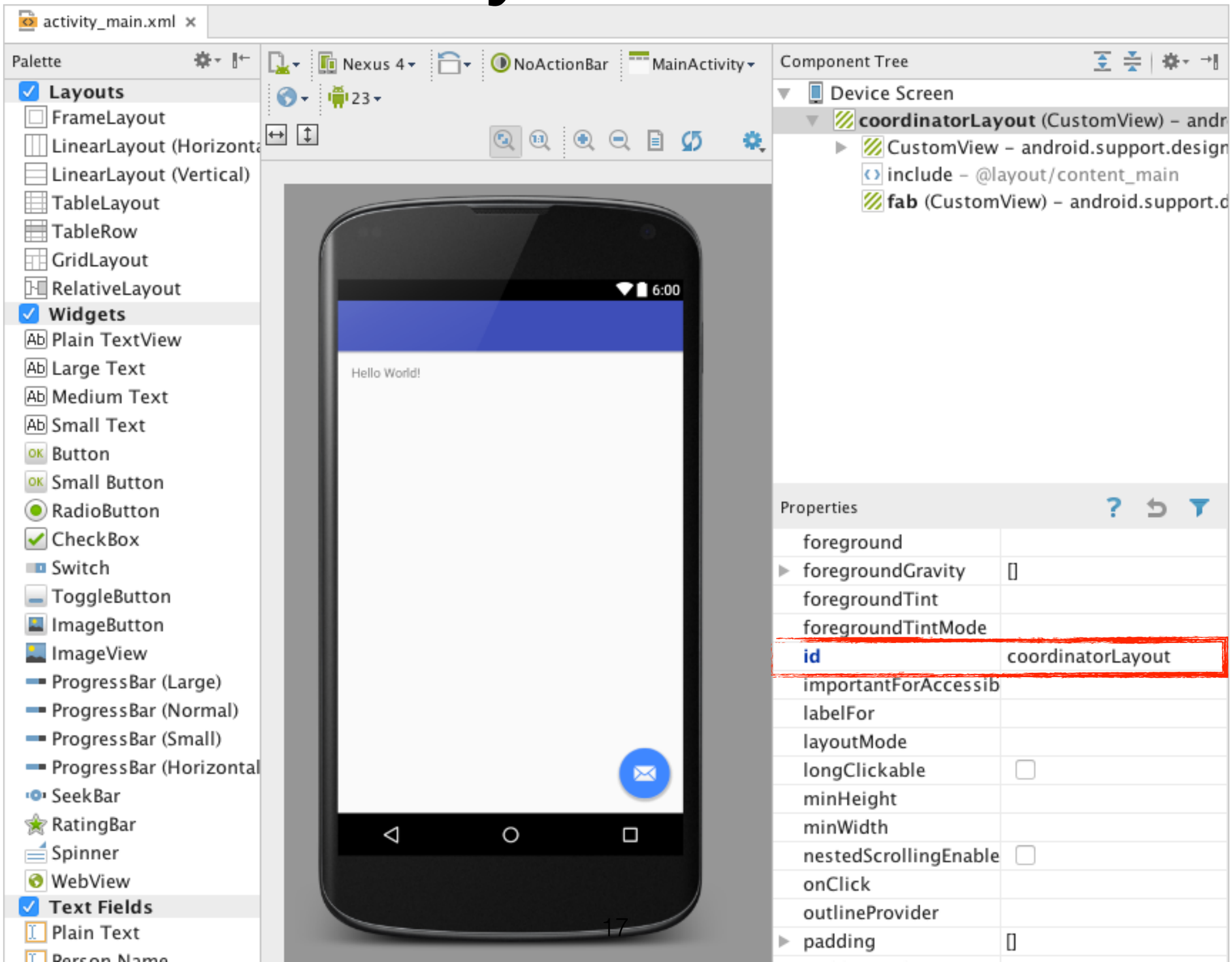
styles.xml öffnen - Open editor



Variante 2

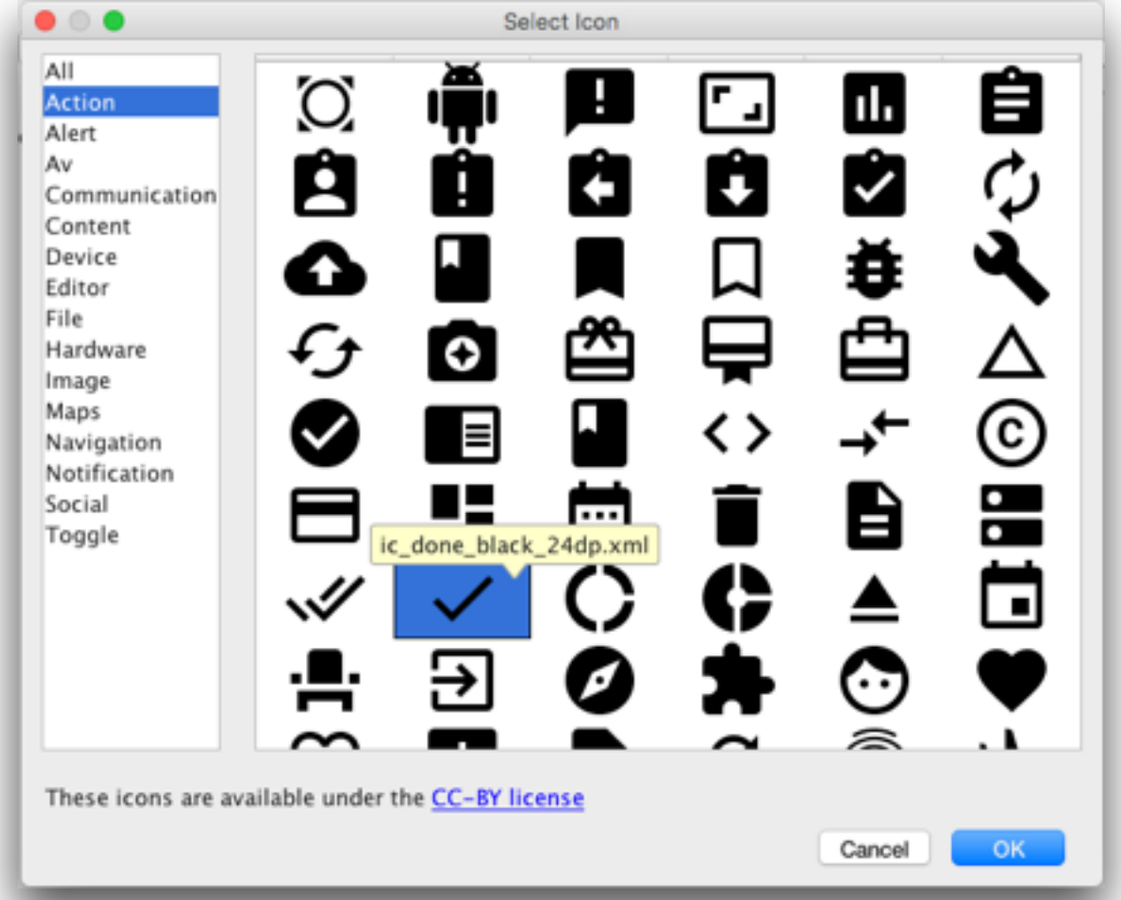

Resources
Color
bright_foreground_disa...
bright_foreground_inver...
bright_foreground_inver...
bright_foreground_mate...
bright_foreground_mate...
button_material_dark
button_material_light
colorAccent
colorPrimary
colorPrimaryDark
design_fab_shadow_end...
design_fab_shadow_mi...
design_fab_shadow_sta...
design_fab_stroke_end_...
design_fab_stroke_end_...
design_fab_stroke_top_...
design_fab_stroke_top_...
design_snackbar_backg...
design_textinput_error_...
Blue accent 200
A: !55 R: 68 G: .38 B: !55
ARGB
448AFF
Name colorAccent
Saving this color will override existing resource colorAccent.
Device Configuration
New Resource
Cancel
OK

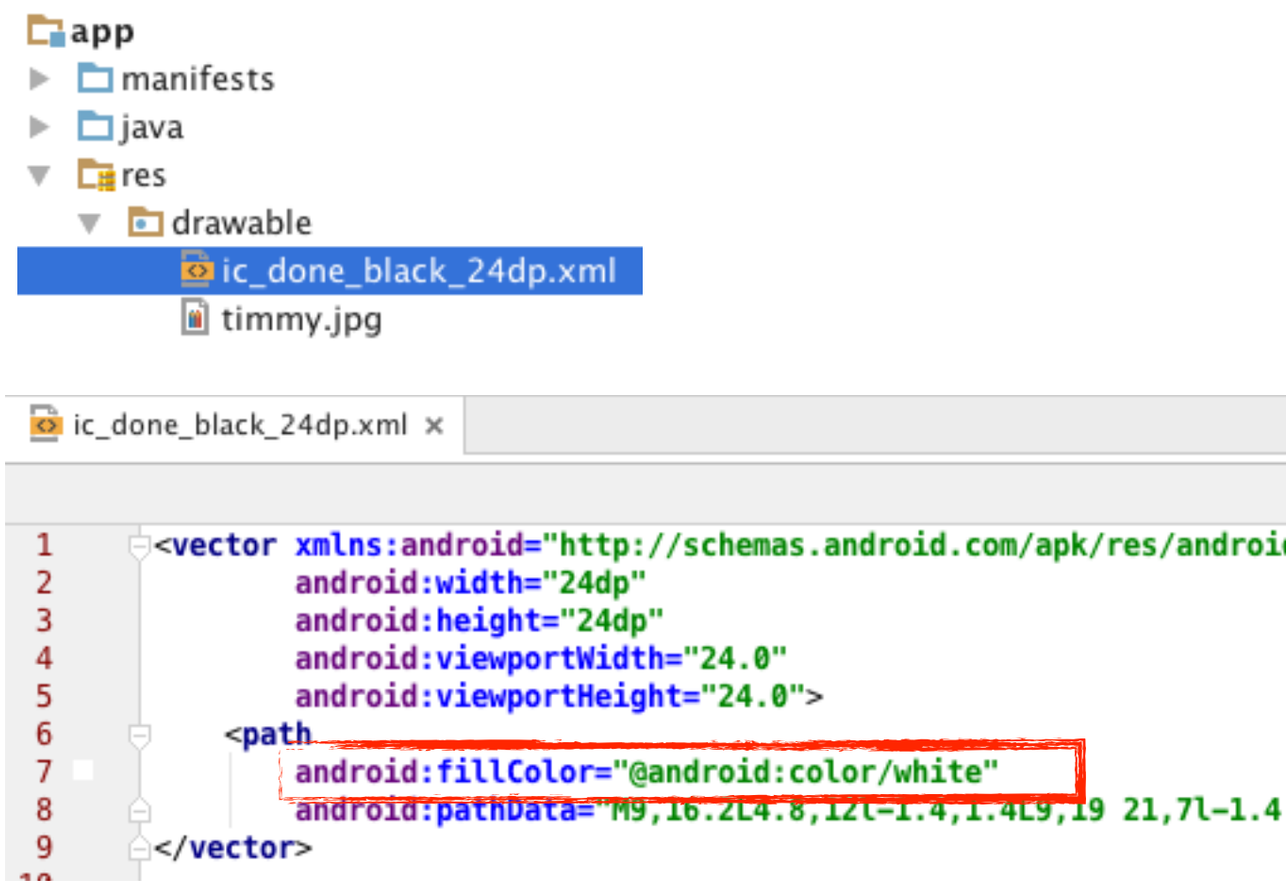
activity_main.xml



Neues Icon

menu_main.xml - New - Vector Asset
- Icon: Choose - ic_done_black_24dp.xml





activity_main.xml

Palette

Nexus 4 | NoActionBar | MainActivity | 23

Component Tree

- Device Screen
  - **coordinatorLayout** (CustomView) – android.support.design
    - CustomView – android.support.design.widget.AppBarLay
    - include – @layout/content_main
    - **fab** (CustomView) – android.support.design.widget.Floa

Properties

importantForAccessibility	
labelFor	
longClickable	☐
maxHeight	
maxWidth	
minHeight	
minWidth	
nestedScrollingEnabled	☐
onClick	
outlineProvider	
▶ padding	[]
paddingEnd	
paddingStart	
scaleType	
▶ scrollIndicators	[]
src	@drawable/ic_done_black_24dp
stateListAnimator	
textAlignment	

activity_main.xml

```
<android.support.design.widget.FloatingActionButton
    android:id="@+id/fab"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="top|end"
    android:layout_margin="16dp"
    android:src="@drawable/ic_done_black_24dp" />
```

- Die layout_gravity kann nicht durch die Design-View eingestellt werden, daher wird sie in der Text-View editiert

activity_main.xml

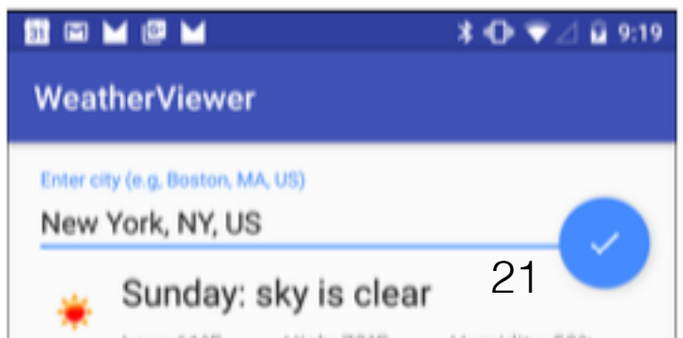
dimens.xml

```xml
<resources>
    <!-- Default screen margins, per the Android Design guidelines. -->
    <dimen name="activity_horizontal_margin">16dp</dimen>
    <dimen name="activity_vertical_margin">16dp</dimen>
    <dimen name="fab_margin">16dp</dimen>
    <dimen name="fab_margin_top">90dp</dimen>
</resources>
```

- Erstellen Sie eine neue Dimension fab_margin_top mit 90dp

```xml
<android.support.design.widget.FloatingActionButton
    android:id="@+id/fab"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="top|end"
    android:layout_marginTop="@dimen/fab_margin_top"
    android:layout_marginEnd="@dimen/fab_margin"
    android:layout_marginBottom="@dimen/fab_margin"
    android:layout_marginStart="@dimen/fab_margin"
    android:src="@drawable/ic_done_black_24dp" />
```

- Ändern Sie die Ränder (margin) wie links angegeben
- Dies ermöglicht dem Button auf der Höhe des noch zu erstellenden EditText-Views zu sein

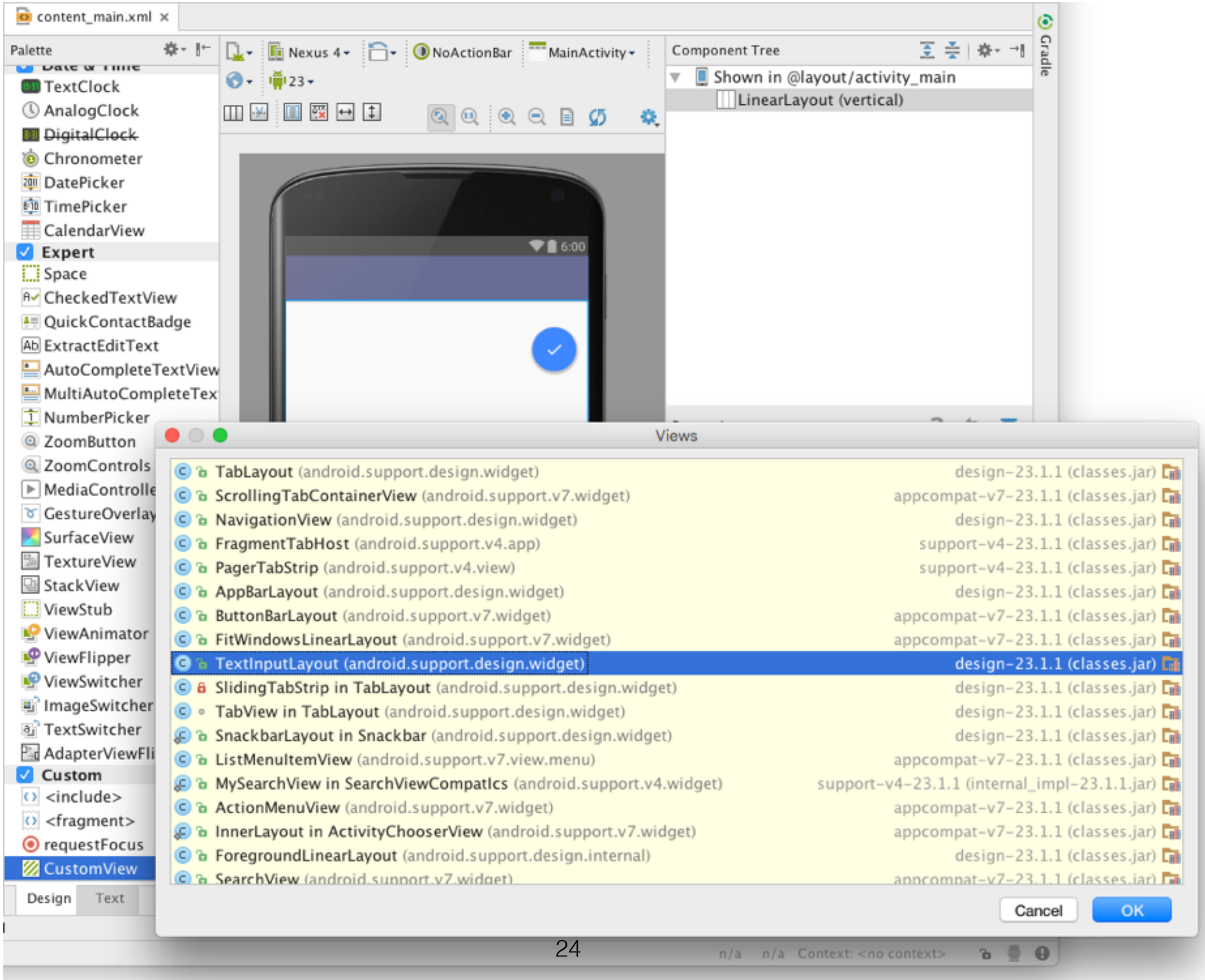


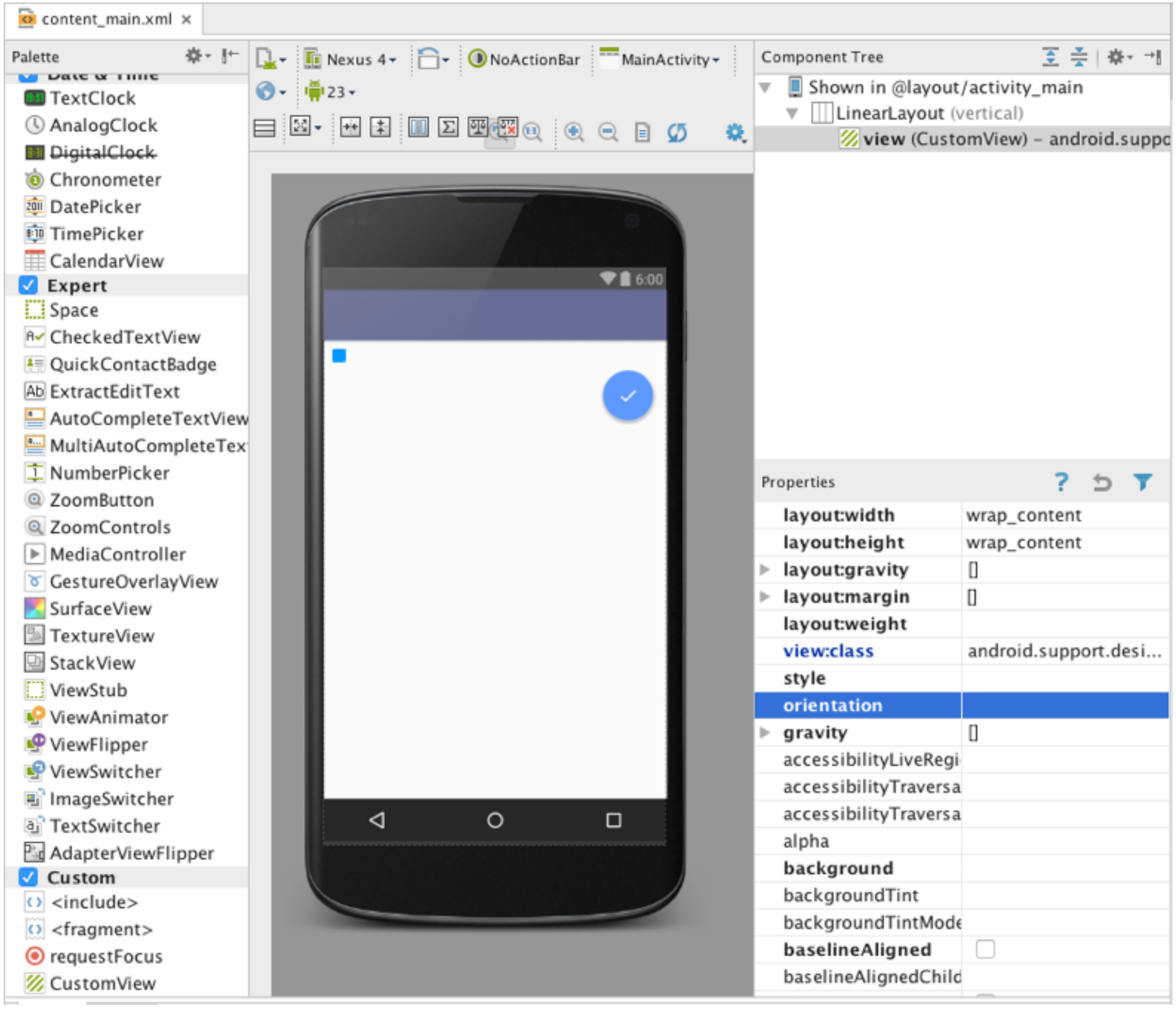
content_main.xml

- Lösche die Hello World - TextView
- Ändere das RelativeLayout zu einem vertikalem LinearLayout
- Klicke in der Palette auf Custom - CustomView und gib im erscheinenden Fenster „TextInputLayout“ ein. Klicke nun den markierten Eintrag aus.
- Bewegen Sie den Cursor über den das LinearLayout im Component Tree bis ein roter Rand erscheint und klicken Sie nun

strings.xml

```
<resources>
    <string name="app_name">WeatherViewer</string>
    <string name="api_key">YOUR_API_KEY</string>
    <string name="web_service_url">http://api.openweathermap.org/data/2.5/forecast/daily?q=</string>
    <string name="invalid_url">Invalid URL</string>
    <string name="weather_condition_image">A graphical representation of the weather conditions</string>
    <string name="high_temp">High: %s</string>
    <string name="low_temp">Low: %s</string>
    <string name="day_description">%1$s: %2$s</string>
    <string name="humidity">Humidity: %s</string>
    <string name="hint_text">Enter city (e.g, Boston, MA, US)</string>
    <string name="read_error">Unable to read weather data</string>
    <string name="connect_error">Unable to connect to OpenWeatherMap.org</string>
</resources>
```


content_main.xml
Palette
TextClock
AnalogClock
DigitalClock
Chronometer
DatePicker
TimePicker
CalendarView
Expert
Space
CheckedTextView
QuickContactBadge
ExtractEditText
AutoCompleteTextView
MultiAutoCompleteTex
NumberPicker
ZoomButton
ZoomControls
MediaControlle
GestureOverlay
SurfaceView
TextureView
StackView
ViewStub
ViewAnimator
ViewFlipper
ViewSwitcher
ImageSwitcher
TextSwitcher
AdapterViewFli
Custom
<include>
<fragment>
requestFocus
CustomView
Design
Text
Nexus 4
NoActionBar
MainActivity
23
6:00
Component Tree
Shown in @layout/activity_main
LinearLayout (vertical)
Gradle
Views
TabLayout (android.support.design.widget) design-23.1.1 (classes.jar)
ScrollingTabContainerView (android.support.v7.widget) appcompat-v7-23.1.1 (classes.jar)
NavigationView (android.support.design.widget) design-23.1.1 (classes.jar)
FragmentTabHost (android.support.v4.app) support-v4-23.1.1 (classes.jar)
PagerTabStrip (android.support.v4.view) support-v4-23.1.1 (classes.jar)
AppBarLayout (android.support.design.widget) design-23.1.1 (classes.jar)
ButtonBarLayout (android.support.v7.widget) appcompat-v7-23.1.1 (classes.jar)
FitWindowsLinearLayout (android.support.v7.widget) appcompat-v7-23.1.1 (classes.jar)
TextInputLayout (android.support.design.widget) design-23.1.1 (classes.jar)
SlidingTabStrip in TabLayout (android.support.design.widget) design-23.1.1 (classes.jar)
TabView in TabLayout (android.support.design.widget) design-23.1.1 (classes.jar)
SnackbarLayout in Snackbar (android.support.design.widget) design-23.1.1 (classes.jar)
ListMenuItemView (android.support.v7.view.menu) appcompat-v7-23.1.1 (classes.jar)
MySearchView in SearchViewCompatIcs (android.support.v4.widget) support-v4-23.1.1 (internal_impl-23.1.1.jar)
ActionMenuView (android.support.v7.widget) appcompat-v7-23.1.1 (classes.jar)
InnerLayout in ActivityChooserView (android.support.v7.widget) appcompat-v7-23.1.1 (classes.jar)
ForegroundLinearLayout (android.support.design.internal) design-23.1.1 (classes.jar)
SearchView (android.support.v7.widget) appcompat-v7-23.1.1 (classes.jar)
Cancel
OK
n/a
n/a
Context: <no context>


content_main.xml
Palette
TextClock
AnalogClock
DigitalClock
Chronometer
DatePicker
TimePicker
CalendarView
Expert
Space
CheckedTextView
QuickContactBadge
ExtractEditText
AutoCompleteTextView
MultiAutoCompleteTex
NumberPicker
ZoomButton
ZoomControls
MediaController
GestureOverlayView
SurfaceView
TextureView
StackView
ViewStub
ViewAnimator
ViewFlipper
ViewSwitcher
ImageSwitcher
TextSwitcher
AdapterViewFlipper
Custom
<include>
<fragment>
requestFocus
CustomView
Nexus 4
NoActionBar
MainActivity
23
6:00
Component Tree
Shown in @layout/activity_main
LinearLayout (vertical)
view (CustomView) – android.suppo
Properties
layout:width
wrap_content
layout:height
wrap_content
layout:gravity
[]
layout:margin
[]
layout:weight
view:class
android.support.desi...
style
orientation
gravity
[]
accessibilityLiveRegi
accessibilityTraversa
accessibilityTraversa
alpha
background
backgroundTint
backgroundTintMode
baselineAligned
baselineAlignedChild

EditText zum TextInputLayout hinzufügen

1

```
droid.support.design.widget.TextInputLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/view" />
```

```
<android.support.design.widget.TextInputLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/view" >

</android.support.design.widget.TextInputLayout>
```

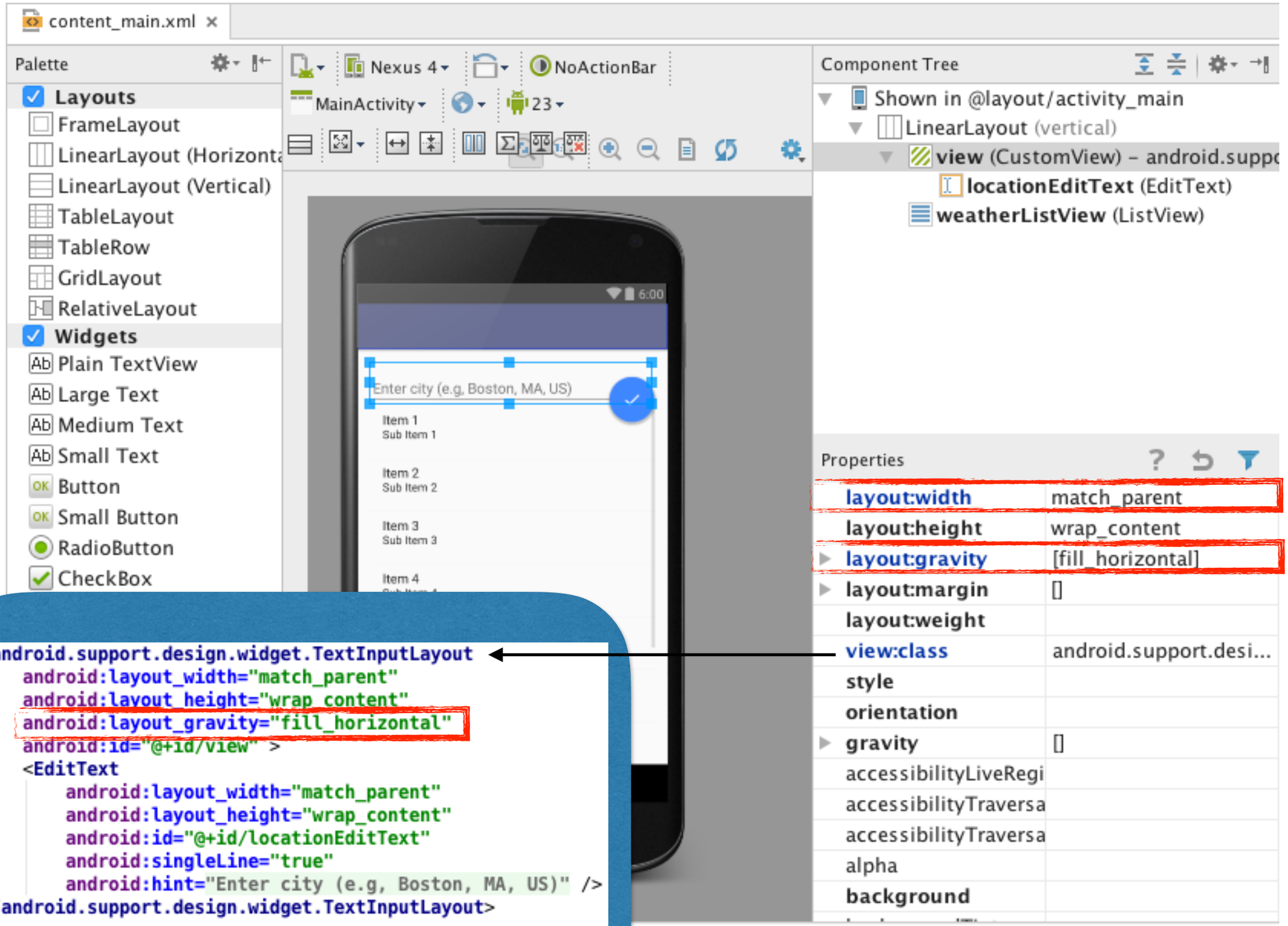
2

```
<android.support.design.widget.TextInputLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/view" >
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</android.support.design.widget.TextInputLayout>
```

3

id	locationEditText
singleLine	☑
hint	@string/hint_text

Enter city (e.g. Boston, MA, US)




```
<android.support.design.widget.TextInputLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="fill_horizontal"
    android:id="@+id/view" >
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/locationEditText"
        android:singleLine="true"
        android:hint="Enter city (e.g, Boston, MA, US)" />
</android.support.design.widget.TextInputLayout>

<ListView
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:id="@+id/weatherListView"
    android:layout_weight="1" />
```

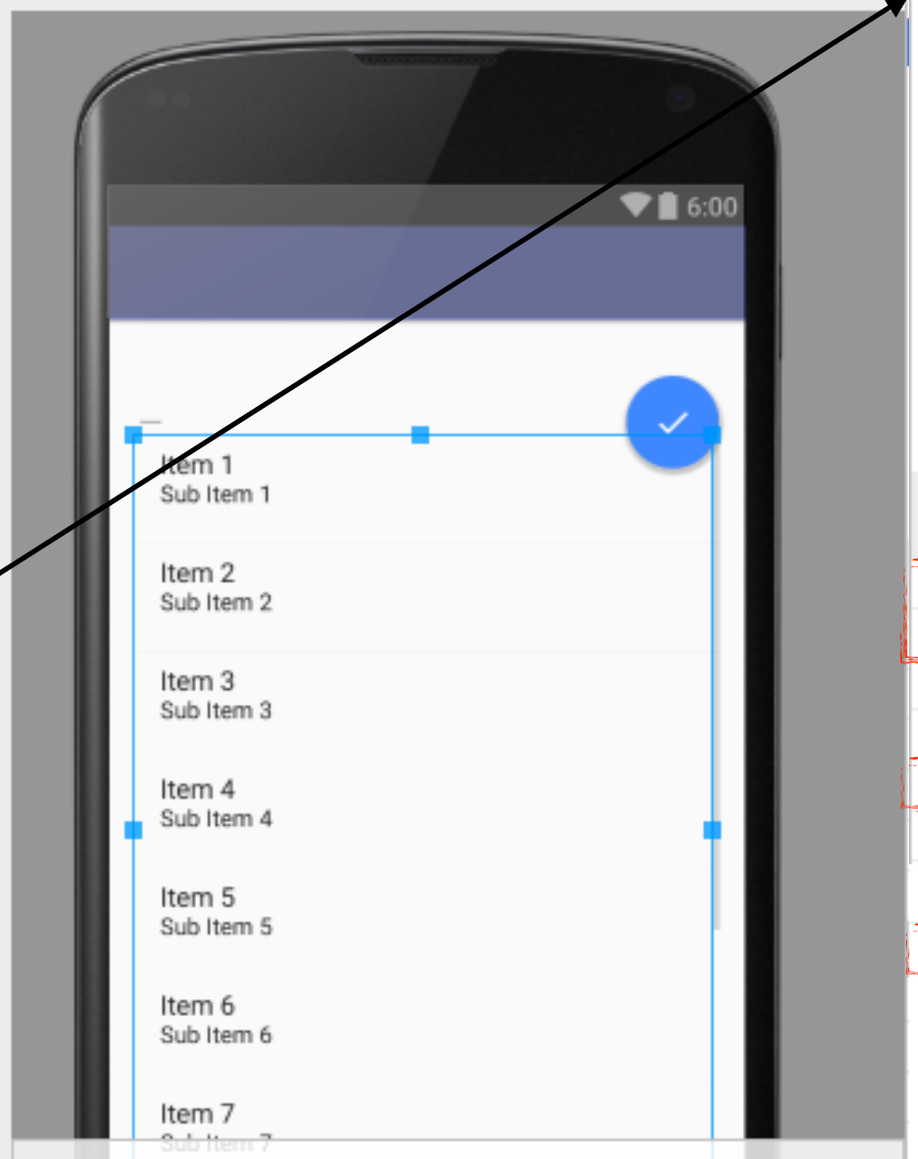
http://developer.samsung.com/technical-doc/view.do?v=T0000000012

content_main.xml

Palette

- Text Fields
- Plain Text
- Person Name
- Password
- Password (Numeric)
- E-mail
- Phone
- Postal Address
- Multiline Text
- Time
- Date
- Number
- Number (Signed)
- Number (Decimal)
- **Containers**
- RadioGroup
- ListView
- GridView
- ExpandableListView
- ScrollView
- HorizontalScrollView
- SearchView
- TabHost
- ~~SlidingDrawer~~
- ~~Gallery~~
- VideoView
- ~~TwoLineListItem~~
- DialerFilter
- **Date & Time**
- TextClock
- AnalogClock
- ~~DigitalClock~~
- Chronometer
- DatePicker
- TimePicker

Nexus 4 | NoActionBar | MainActivity | 23



Rendering Problems

The graphics preview in the layout editor may not be accurate:

- Paint.setShadowLayer is not supported. (Ignore for this session)

Ignore all fidelity warnings for this session

Component Tree

- Shown in @layout/activity_main
- LinearLayout (vertical)
- **view** (CustomView) – android.support.c
- **locationEditText** (EditText)
- **weatherListView** (ListView)

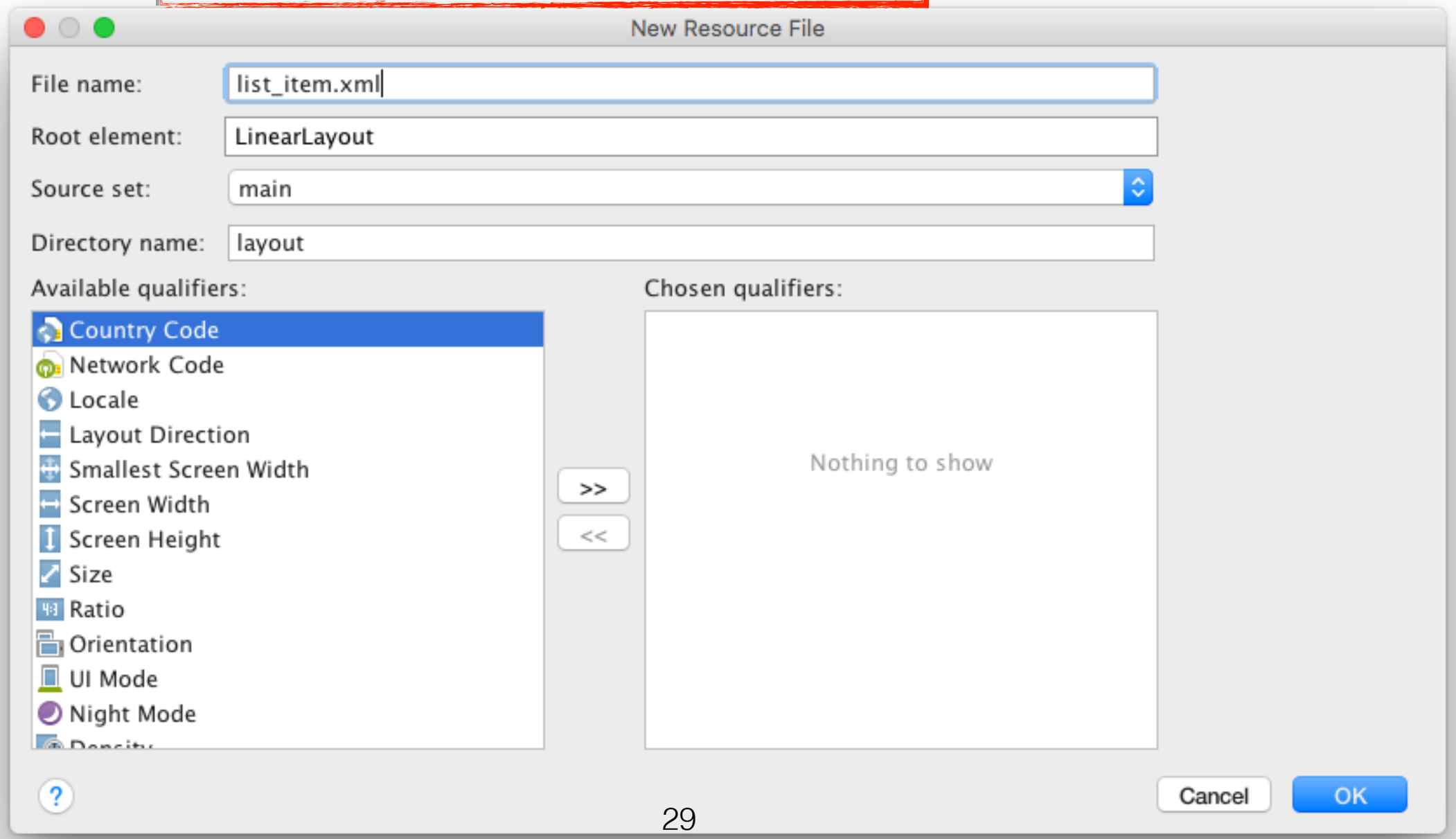
Properties

layout:width	match_parent
layout:height	0dp
layout:gravity	[]
layout:margin	[]
layout:weight	1
style	
id	weatherListView

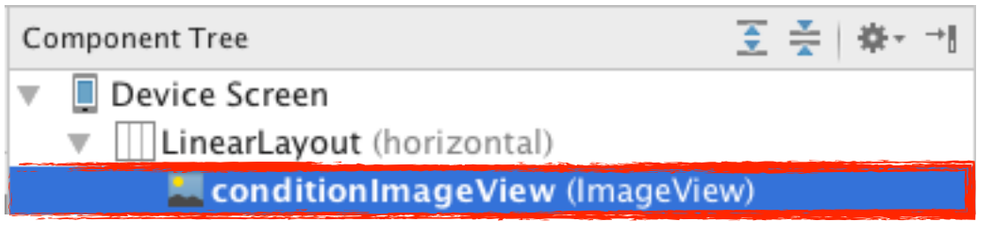
list_item.xml

app/layout - New - Layout resource file

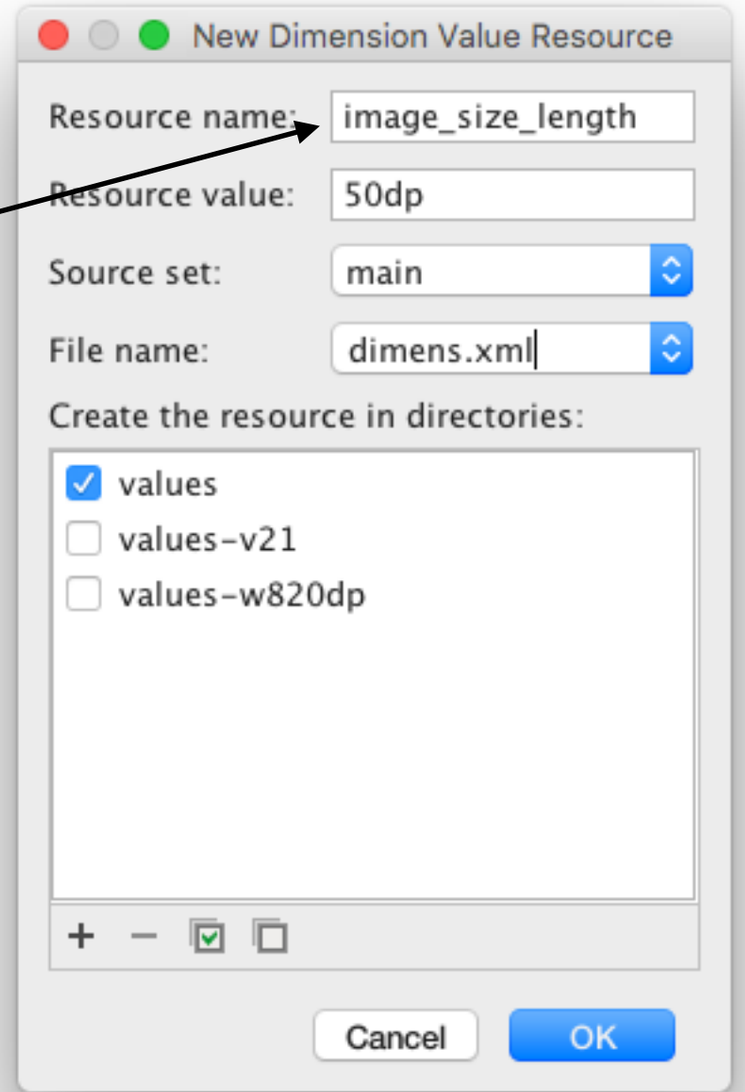
orientation	horizontal

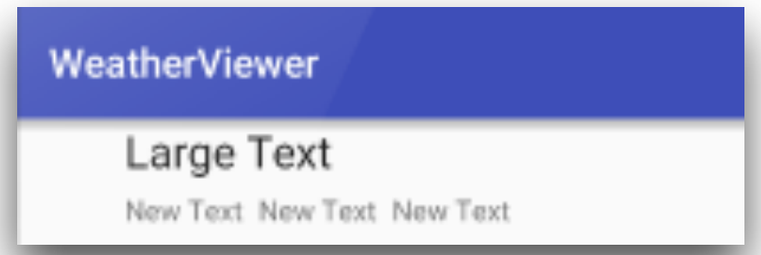


list_item.xml



layout:width	@dimen/image_size_length
layout:height	match_parent
contentDescription	@string/weather_condition_image
id	conditionImageView
scaleType	fitCenter





GridLayout

Ein Standard-Abstand zwischen den Zellen wird eingehalten

Component Tree

- Device Screen
  - LinearLayout (horizontal)
    - **conditionImageView** (ImageView)
    - GridLayout (2, 3, horizontal)

layout:width	0**dp**
layout:height	match_parent
layout:weight	1
columnCount	3
rowCount	2
useDefaultMargins	☑

Component Tree

- Device Screen
  - LinearLayout (horizontal)
    - **conditionImageView** (ImageView)
    - GridLayout (2, 3, horizontal)
      - **dayTextView** (TextView) – "Large Text"
      - **lowTextView** (TextView) – "New Text"
      - **hiTextView** (TextView) – "New Text"
      - **humidityTextView** (TextView) – "New Text"

Large Text

layout:column	0
layout:columnSpan	3
id	dayTextView

PlainText

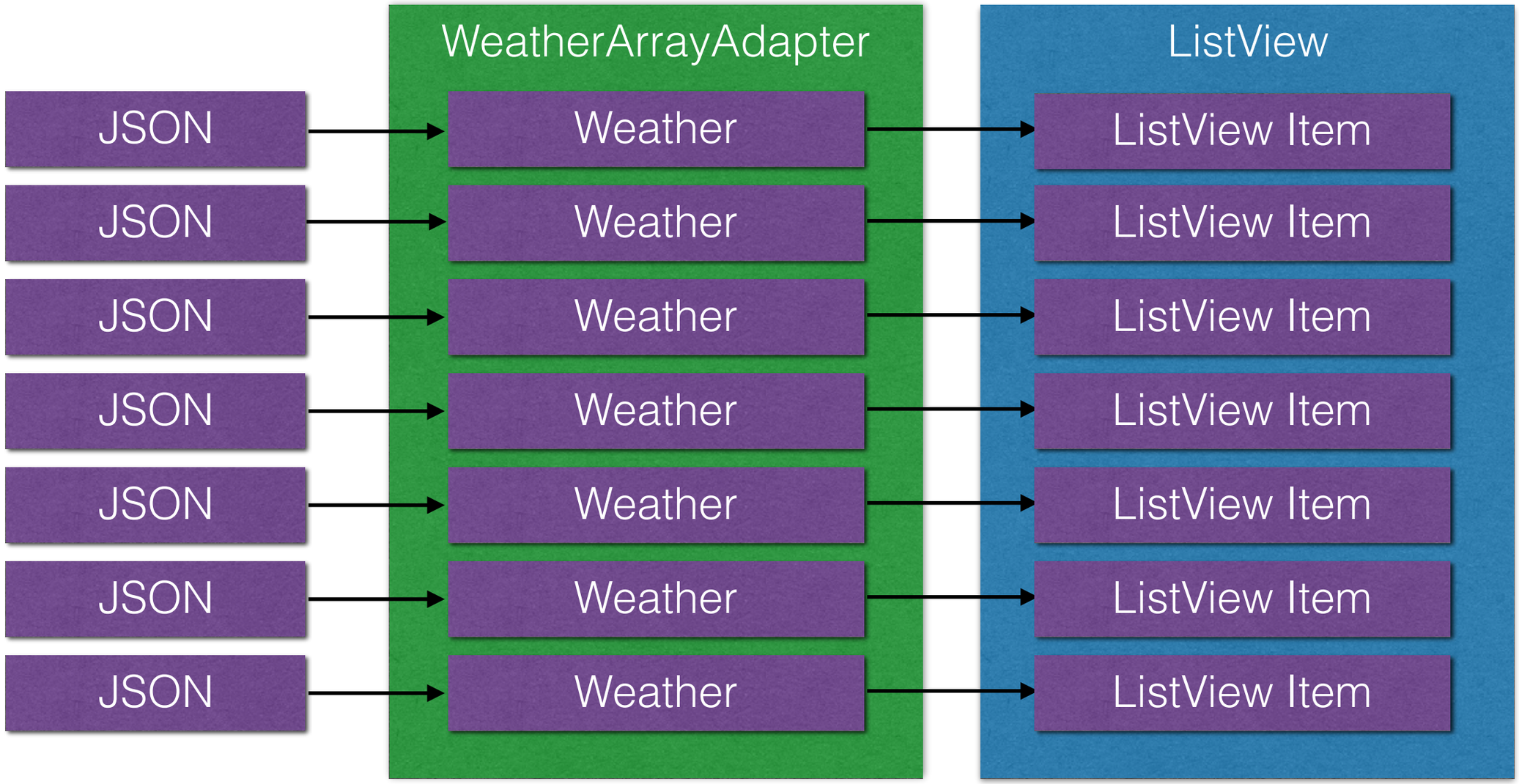
layout:column	0
layout:row	1
layout:rowWeight	1
id	lowTextView

PlainText

layout:column	1
layout:row	1
layout:rowWeight	1
id	hiTextView

PlainText

layout:column	2
layout:row	1
layout:rowWeight	1
id	humidityTextView


WeatherArrayAdapter
ListView
JSON
JSON
JSON
JSON
JSON
JSON
Weather
Weather
Weather
Weather
Weather
Weather
ListView Item
ListView Item
ListView Item
ListView Item
ListView Item
ListView Item

Weather.java 1

```
public class Weather {

    public final String dayOfWeek;
    public final String minTemp;
    public final String maxTemp;
    public final String humidity;
    public final String description;
    public final String iconURL;

    public Weather(long timeStamp, double minTemp, double maxTemp,
                   double humiditiy, String description, String iconName) {

        //NumberFormat to format double temperatures rounded to integers
        NumberFormat numberFormat = NumberFormat.getInstance()    ;
        numberFormat.setMaximumFractionDigits(0);

        this.dayOfWeek = convertTimeStampToDay(timeStamp);
        this.minTemp = numberFormat.format(minTemp) + "\u00B0C";
        this.maxTemp = numberFormat.format(maxTemp) + "\u00B0C";
        this.humidity = NumberFormat.getPercentInstance().format(humiditiy / 100.0);
        this.description = description;
        this.iconURL = "http://openweathermap.org/img/w/" + iconName + ".png";
    }
}
```

Create method 'convertTimeStampToDay'

Create switch statement

Weather.java

```
// convert timestamp to a day's name (e.g. Monday, Tuesday, ..)
private static String convertTimeStampToDay(long timeStamp) {
    Calendar calendar = Calendar.getInstance();
    calendar.setTimeInMillis(timeStamp * 1000);
    TimeZone tz = TimeZone.getDefault(); // get device's time zone

    // adjust time for device's time zone
    calendar.add(Calendar.MILLISECOND, tz.getOffset(calendar.getTimeInMillis()));

    // SimpleDateFormat that returns the day's name
    SimpleDateFormat dateFormatter = new SimpleDateFormat("EEEE");
    return dateFormatter.format(calendar.getTime());
}
```

WeatherArrayAdapter.java 1

```
public class WeatherArrayAdapter extends ArrayAdapter<Weather> {

    // class for reusing views as list items scroll off onto the screen
    private static class ViewHolder {
        ImageView conditionImageView;
        TextView dayTextView;
        TextView lowTextView;
        TextView hiTextView;
        TextView humidityTextView;
    }
```

- Für jedes vom Adapter erstellte ListView-Item (!) wird ein ViewHolder-Objekt erstellt. Bei einem existierenden ListItem, welches wiederverwendet wird, wird auch das ViewHolder-Objekt wiederverwendet.

ViewHolder pattern

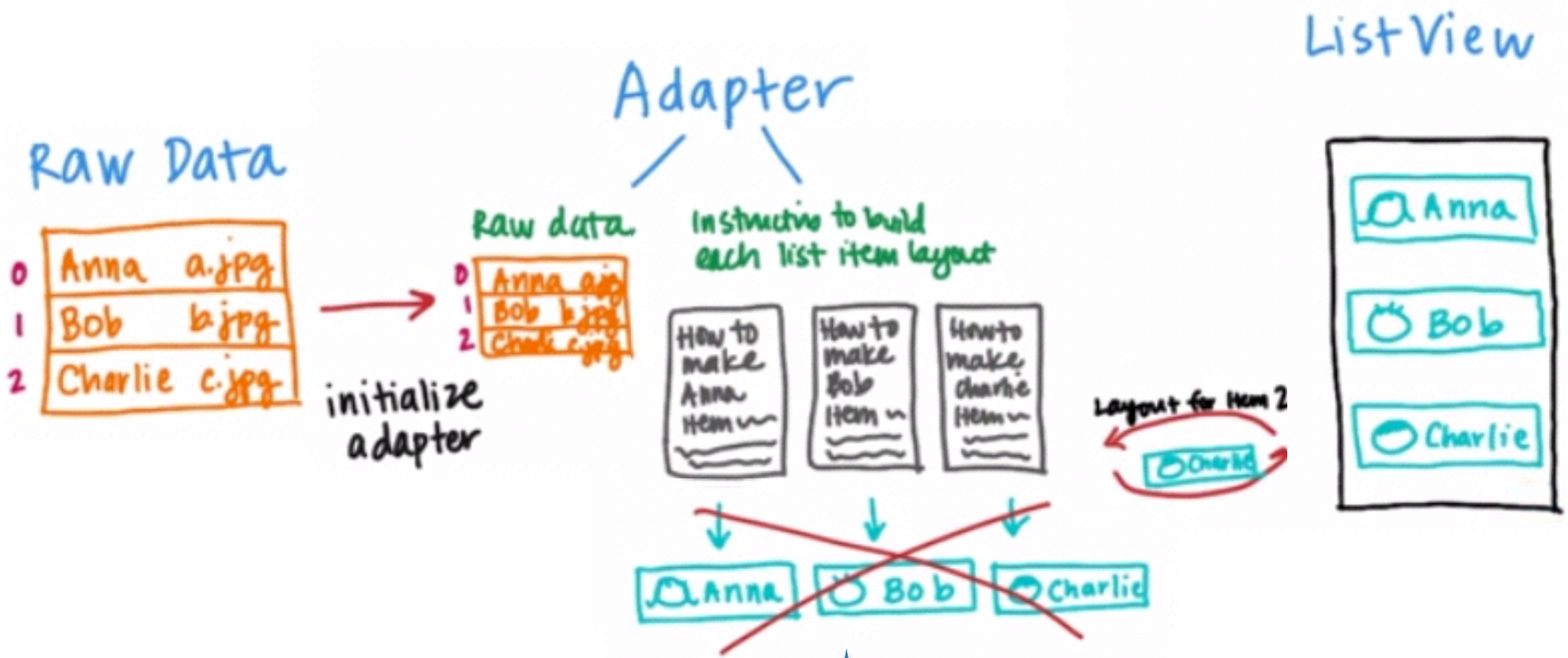
- A ViewHolder implementation allows to avoid the findViewById() method in an adapter.

- A ViewHolder class is typically a static inner class in your adapter which holds references to the relevant views in your layout. This reference is assigned to the row view as a tag via the setTag() method.

- If we receive a rowView object, we can get the instance of the ViewHolder via the getTag() method and assign the new attributes to the views via the ViewHolder reference.

- While this sounds complex this is approximately 15 % faster then using the findViewById() method.

http://www.vogella.com/tutorials/AndroidListView/article.html
http://developer.android.com/training/improving-layouts/smooth-scrolling.html



Die TextViews werden erst erstellt, wenn sie von der ListView angefragt werden

WeatherArrayAdapter.java 2

```
// stores already downloaded Bitmaps for reuse
private Map<String, Bitmap> bitmaps = new HashMap<>();

// constructor to initilize superclass inherited members
public WeatherArrayAdapter(Context context, List<Weather> forecast) {
    super(context, -1, forecast);
}
```

- context … the Activity in which the ListView is displayed

- -1 … a layout resource id for a **layout that contains a TextView** in which a ListView item's data is displayed. The argument -1 indicates that we use a **custom layout** in this app, so we can display more than just one TextView

- forecast … the List of data to display

WeatherArrayAdapter.java

getView() 1

gibt die View zurück, die für jedes Item erstellt wird

^O – getView ⏎

```
// creates the custom view for the ListView's items
@Override
public View getView(int position, View convertView, ViewGroup parent) {

    View rowView = convertView;

    // get Weather object for this specified ListView position
    Weather day = getItem(position);

    ViewHolder viewHolder; // object that references's list item's views

    // check for reusable ViewHolder from ListView item that scrolled
    // offscreen; otherwise create a new ViewHolder
    if (rowView == null) {  // no reusable ViewHolder, so create one
        viewHolder = new ViewHolder();
        LayoutInflater inflater = LayoutInflater.from(getContext());
        rowView = inflater.inflate(R.layout.list_item, parent, false);
        viewHolder.conditionImageView =
                (ImageView) rowView.findViewById(R.id.conditionImageView);
        viewHolder.dayTextView =
                (TextView) rowView.findViewById(R.id.dayTextView);
        viewHolder.lowTextView =
                (TextView) rowView.findViewById(R.id.lowTextView);
        viewHolder.hiTextView =
                (TextView) rowView.findViewById(R.id.hiTextView);
        viewHolder.humidityTextView =
                (TextView) rowView.findViewById(R.id.humidityTextView);
        rowView.setTag(viewHolder);  // assign the data to the item as a tag
    } else { // reuse existing ViewHolder stored as the list item's tag
        viewHolder = (ViewHolder) rowView.getTag();
    }
```

WeatherArrayAdapter.java

getView() 2

```
// if weather condition icon already downloaded use it;
// otherwise, download icon in a separate thread
if (bitmaps.containsKey(day.iconURL)) {
    viewHolder.conditionImageView.setImageBitmap(bitmaps.get(day.iconURL));
} else {
    // download and display weather condition image
    Log.d(LOG_TAG, "day.iconURL = " + day.iconURL);
    new LoadImageTask(viewHolder.conditionImageView).execute(day.iconURL);
}

// get other data from Weather object and place into views
Context context = getContext(); // for loading String resources
viewHolder.dayTextView.setText("{day.dayOfWeek}: {day.description}");
viewHolder.lowTextView.setText(
        "Low: {day.minTemp}");
viewHolder.hiTextView.setText(
        "High: {day.maxTemp}");
viewHolder.humidityTextView.setText(
        "Humidity: {day.humidity}");

return rowView; // return completed list item to display
}
```

WeatherArrayAdapter.java

LoadImageTask

```
// AsyncTask to load weather conditions icons in a separate thread
private class LoadImageTask extends AsyncTask<String, Void, Bitmap> {
    private ImageView imageView; // displays the thumbnail

    // store ImageView on which to set the download Bitmap
    public LoadImageTask(ImageView imageView) {
        this.imageView = imageView;
    }

    // load image: params[0] is the String URL representing the image
    @Override
    protected Bitmap doInBackground(String... params) {...}

    // set weather image in list item
    @Override
    protected void onPostExecute(Bitmap bitmap) {
        imageView.setImageBitmap(bitmap);
    }
}
```

```java
// store ImageView on which to set the download Bitmap
@Override
protected Bitmap doInBackground(String... params) {
    Bitmap bitmap = null;
    HttpURLConnection connection = null;

    try {
        URL url = new URL(params[0]); // create URL for image

        // open an HttpURLConnection, get its InputStream
        // and download the image
        connection = (HttpURLConnection) url.openConnection();

        try (InputStream inputStream = connection.getInputStream()) {
            bitmap = BitmapFactory.decodeStream(inputStream);
            bitmaps.put(params[0], bitmap); // cache for later use
        } catch (Exception e) {
            e.printStackTrace();
        }
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        connection.disconnect();
    }
    return bitmap;
}
```

Software Engineering Observation 7.1

Every time an `AsyncTask` *is required, you must create a new object of your* `AsyncTask` *type—each* `AsyncTask` *can be executed only once.*

MainActivity.java 1

```
public class MainActivity extends AppCompatActivity {

    private static final String LOG_TAG = MainActivity.class.getSimpleName();

    // List of Weather objects representing the forecast
    private List<Weather> weatherList = new ArrayList<>();

    // ArrayAdapter for binding Weather objects to a ListView
    private WeatherArrayAdapter weatherArrayAdapter;
    private ListView weatherListView;
```

MainActivity.java 2 - onCreate() 1

```
// configure Toolbar, ListView and FAB
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    // autogenerated code to inflate layout and configure Toolbar
    setContentView(R.layout.activity_main);
    Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
    setSupportActionBar(toolbar);

    // create ArrayAdapter to bind weatherList to the weatherListView
    weatherListView = (ListView) findViewById(R.id.weatherListView);
    weatherArrayAdapter = new WeatherArrayAdapter(this, weatherList);
    weatherListView.setAdapter(weatherArrayAdapter);

    weatherListView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
            Toast.makeText(getBaseContext(), "Item clicked", Toast.LENGTH_SHORT).show();
            Snackbar.make(findViewById(R.id.coordinatorLayout), "item clicked", Snackbar.LENGTH_SHORT).show();
        }
    });
```

… see next slide

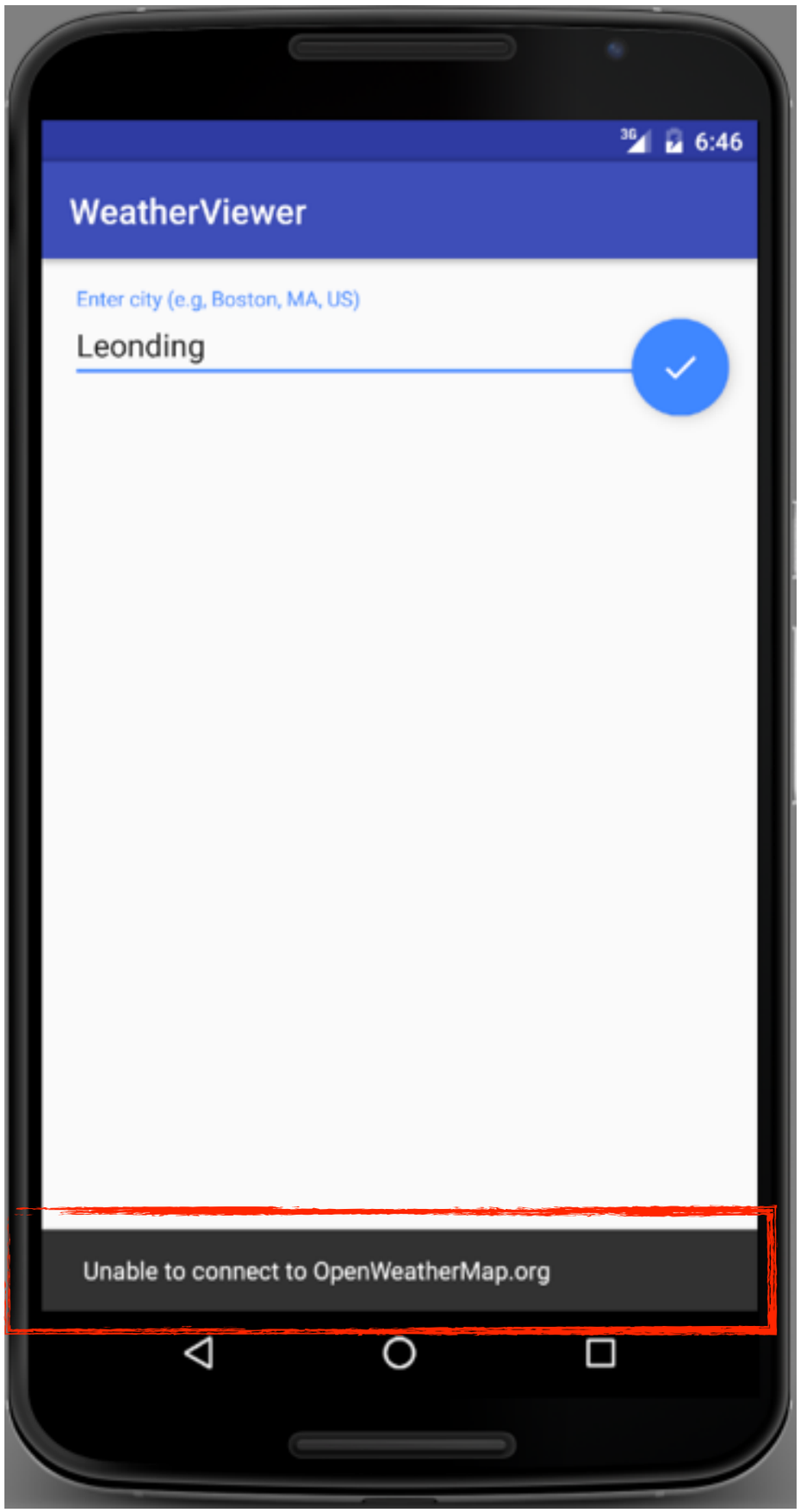
```
}
```

MainActivity.java 3 - onCreate() 2

```
// Configure FAB to hide keyboard and initiate web service request
FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab);
fab.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        // get next from locationEditText and create web service URL
        EditText locationEditText = (EditText) findViewById(R.id.locationEditText);
        URL url = createURL(locationEditText.getText().toString());

        // hide keyboard and initiate a GetWeatherTask to download
        // weather data from OpenWeatherMap.org in a separate thread
        if (url != null) {
            dismissKeyboard(locationEditText);
            GetWeatherTask getLocalWeatherTask = new GetWeatherTask();
            getLocalWeatherTask.execute(url);
        } else {
            Snackbar.make(findViewById(R.id.coordinatorLayout),
                    "Invalid URL", Snackbar.LENGTH_LONG).show();
        }
                R.string.invalid_url
    }
});
```

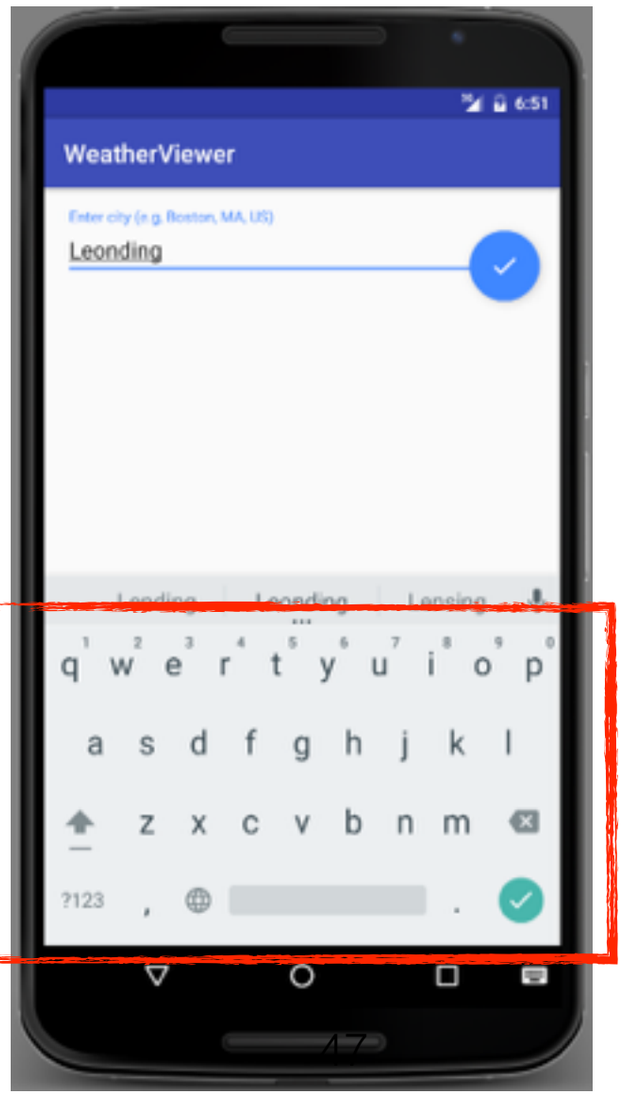
Snackbar

```
Snackbar.make(findViewById(
        R.id.coordinatorLayout),
        R.string.connect_error,
        Snackbar.LENGTH_LONG
).show();
```



MainActivity.java 4

```
private void dismissKeyboard(View view) {
    InputMethodManager imm =
            (InputMethodManager) getSystemService(Context.INPUT_METHOD_SERVICE);
    imm.hideSoftInputFromWindow(view.getWindowToken(), 0);
}
```



MainActivity.java 5

```
// create openweathermap.org web service URL using city
private URL createURL(String city) {
    String apiKey = getString(R.string.api_key);
    String baseUrl = getString(R.string.web_service_url);

    try {
        // create URL for specified city and metric units (Celsius)
        String urlString = baseUrl + URLEncoder.encode(city, "UTF-8") +
                "&units=metric&cnt=16&APPID=" + apiKey;
        return new URL(urlString);
    } catch (Exception e) {
        e.printStackTrace();
    }

    return null; // URL was malformed

}
```

- imperial … Fahrenheit, metric … Celsius, standard … Kelvin
- cnt=16 … Vorhersage für die nächsten 16 Tage
- deutsche Wettervorhersage

```
String urlString = baseUrl + URLEncoder.encode(city, "UTF-8") +
        "&units=metric&lang=de&cnt=16&APPID=" + apiKey;
```

MainActivity.java 6 - GetWeatherTask 1

```
// makes the REST web service call to get weather data and
// saves the data to a local HTML file
private class GetWeatherTask extends AsyncTask<URL, Void, JSONObject> {

    @Override
    protected JSONObject doInBackground(URL... params) {
        HttpURLConnection connection = null;

    try {
```

… see next slide

```
    } catch (Exception e) {
        Snackbar.make(findViewById(R.id.coordinatorLayout),
                R.string.connect_error, Snackbar.LENGTH_LONG).show();
        e.printStackTrace();
    } finally {
        connection.disconnect(); // close the HttpURLConnection
    }

    return null;
}
```

Eingangs-Parameter

Ausgangs-Parameter

zur Anzeige des Fortschritts - onProgressUpdate()

MainActivity.java 7 - GetWeatherTask 2

```
connection = (HttpURLConnection) params[0].openConnection();
int response = connection.getResponseCode();

if (response == HttpURLConnection.HTTP_OK) {
    StringBuilder builder = new StringBuilder();

    try (BufferedReader reader = new BufferedReader(
            new InputStreamReader(connection.getInputStream()))) {

        String line;

        while ((line = reader.readLine()) != null) {
            builder.append(line);
        }
    } catch (IOException e) {
        Snackbar.make(findViewById(R.id.coordinatorLayout),
                R.string.read_error, Snackbar.LENGTH_LONG).show();
        e.printStackTrace();
    }

    return new JSONObject(builder.toString());
} else {
    Snackbar.make(findViewById(R.id.coordinatorLayout),
            R.string.connect_error, Snackbar.LENGTH_LONG).show();
}
```

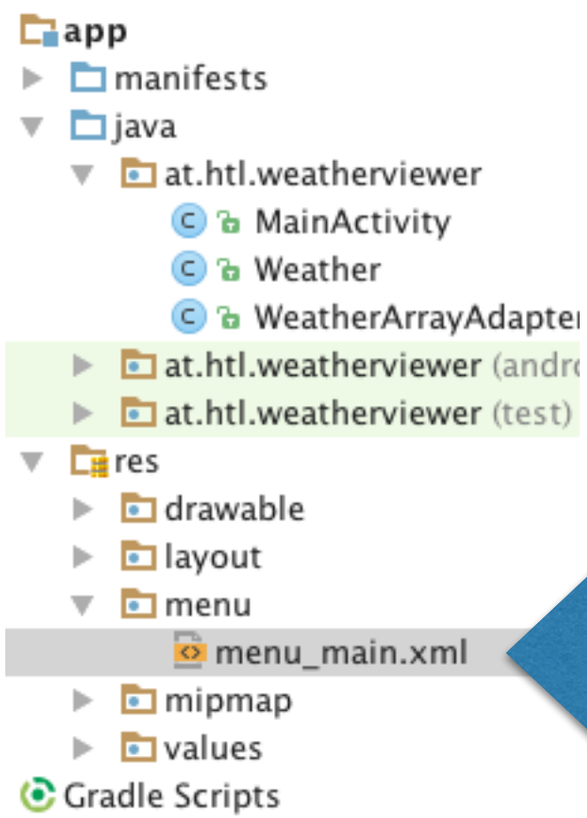
MainActivity.java 8 - GetWeatherTask 3

```
        // process JSON response and update ListView
        @Override
        protected void onPostExecute(JSONObject weather) {
            convertJSONtoArrayList(weather); // repopulate weatherList
            weatherArrayAdapter.notifyDataSetChanged(); // rebind to ListView
            weatherListView.smoothScrollToPosition(0); // scroll to top
        }
}
```

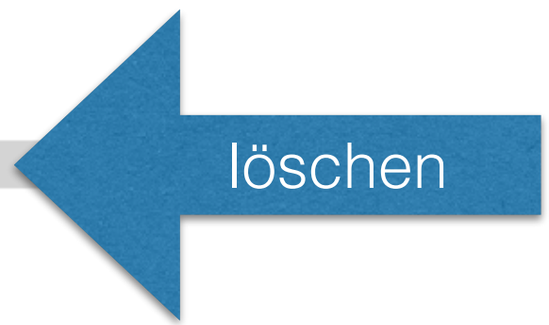
MainActivity.java 9

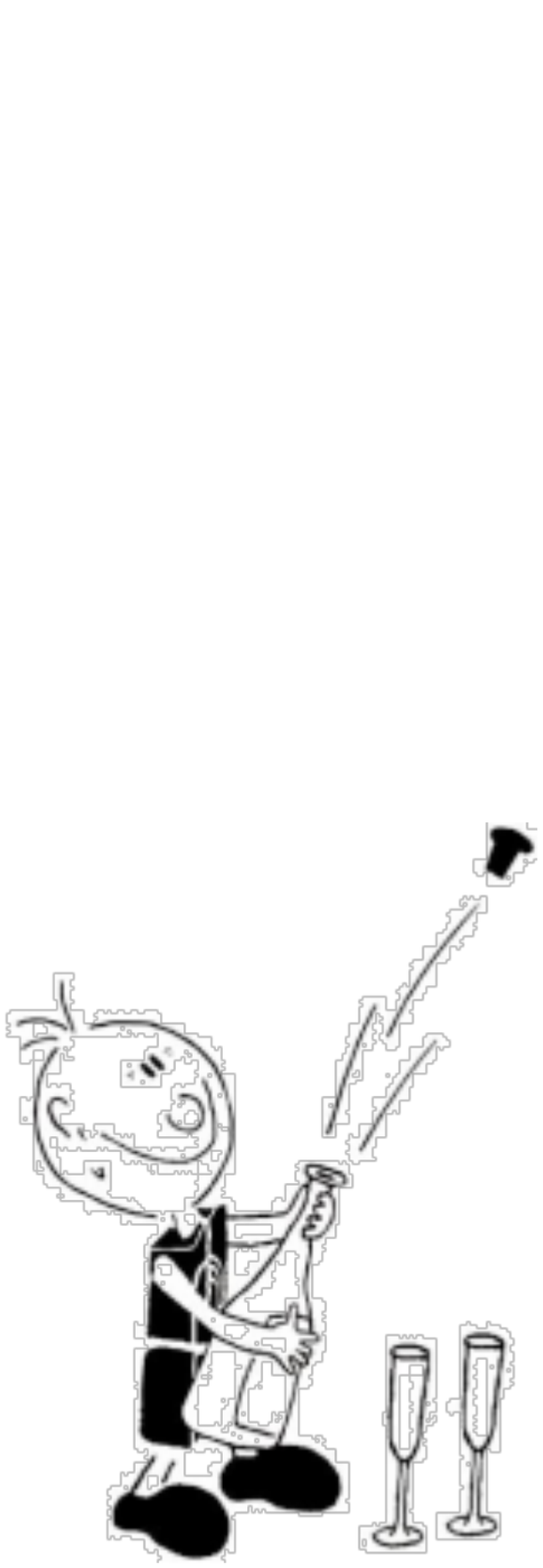
```
// create Weather objects from JSONObject containing the forecast
private void convertJSONtoArrayList(JSONObject forecast) {
    weatherList.clear(); // clear old weather data

    try {
        // get forecast's "list" JSONArray
        JSONArray list = forecast.getJSONArray("list");

        // convert each element of list to a Weather object
        for (int i = 0; i < list.length(); ++i) {
            JSONObject day = list.getJSONObject(i); // get one day's data

            // get the day's temperatures ("temp") JSONObject
            JSONObject temperatures = day.getJSONObject("temp");

            // get day's "weather" JSONObject for the description and icon
            JSONObject weather =
                    day.getJSONArray("weather").getJSONObject(0);

            // add new Weather object to weatherList
            weatherList.add(new Weather(
                    day.getLong("dt"), // date/time timestamp
                    temperatures.getDouble("min"), // minimum temperature
                    temperatures.getDouble("max"), // maximum temperature
                    day.getDouble("humidity"), // percent humidity
                    weather.getString("description"), // weather conditions
                    weather.getString("icon"))); // icon name
        }
    } catch (JSONException e) {
        e.printStackTrace();
    }
}
```

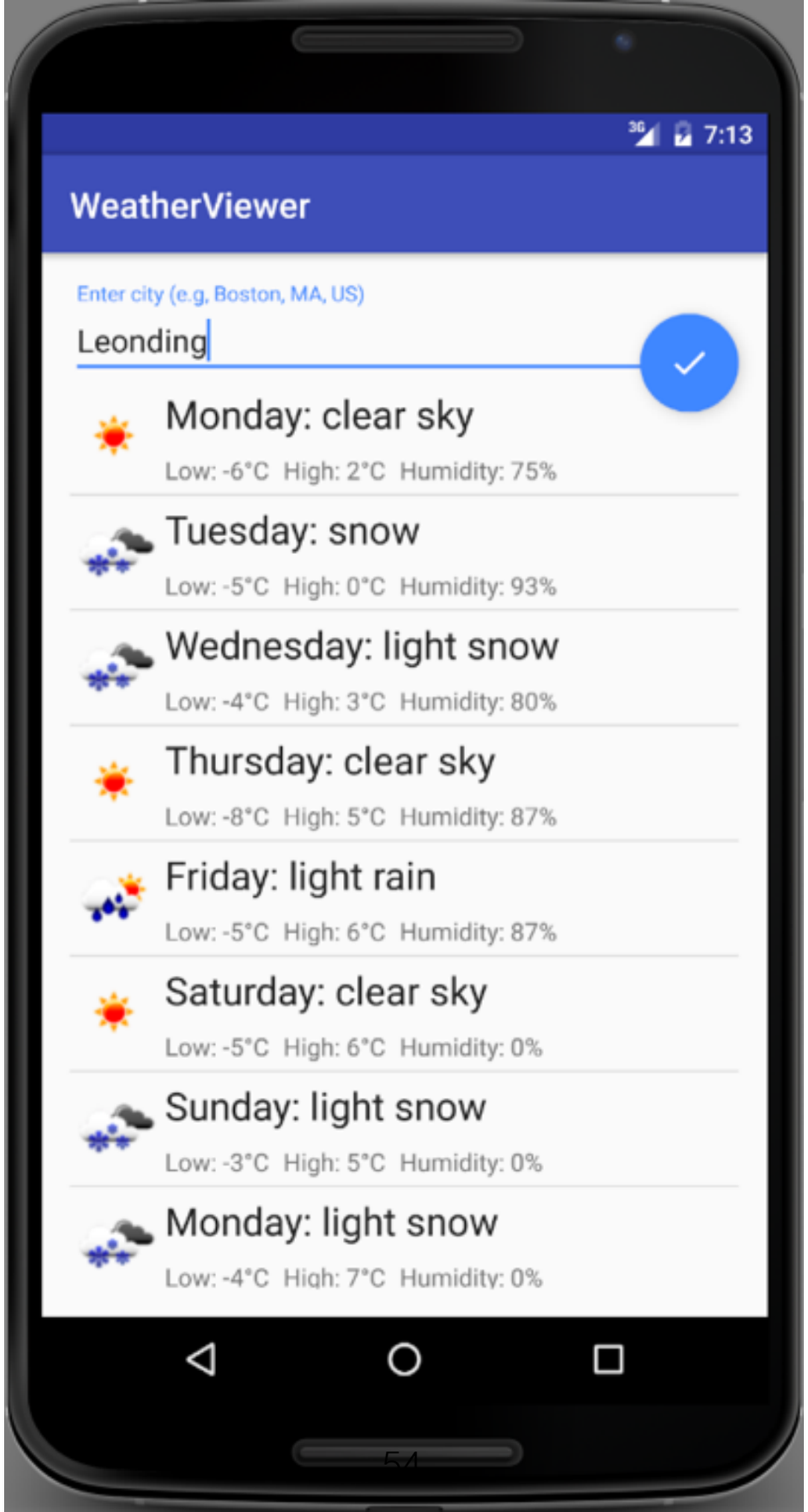
Menu wird nicht benötigt



```
@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);
    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();

    //noinspection SimplifiableIfStatement
    if (id == R.id.action_settings) {
        return true;
    }

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






7:13
WeatherViewer
Enter city (e.g, Boston, MA, US)
Leonding
Monday: clear sky
Low: -6°C High: 2°C Humidity: 75%
Tuesday: snow
Low: -5°C High: 0°C Humidity: 93%
Wednesday: light snow
Low: -4°C High: 3°C Humidity: 80%
Thursday: clear sky
Low: -8°C High: 5°C Humidity: 87%
Friday: light rain
Low: -5°C High: 6°C Humidity: 87%
Saturday: clear sky
Low: -5°C High: 6°C Humidity: 0%
Sunday: light snow
Low: -3°C High: 5°C Humidity: 0%
Monday: light snow
Low: -4°C High: 7°C Humidity: 0%


Werbung

Aufgabe

- Internationalisieren Sie die Ausgabe für Deutsch

- Verwenden Sie die Wetterengine: https://www.wunderground.com/

Noch Fragen?