Introduction to App Development for Android

A first App

Prof. Dr. Paul Rawiel

A first App, that doesn‘t just show the text Hello World, but says hello

- New project with empty Activity
- Project Name: SaySomething
- Package Name: de.cursoufpr.saysomething
- Select Minimum SDK API

→ create project

Project structure as shown before

Important nodes

- java for code
- res for layout

The MainActivity

- Code under the java node in the tree structure of the project
- the layout under the res node

- Class MainActivity inherits from the the class AppCompatActivity
- the method onCreate()
  - executed when the App starts
  - calls onCreate of the parent class
  - Important administrative tasks in the background
  - setContentView defines the content of the screen

Voice output

- Class TextToSpeech
- The MainActivity need an attribute of the type of this class

```
private TextToSpeech tts;
```

In the onCreate method

```
tts = new TextToSpeech(this, this);
```

Voice output with the command

```
tts.speak(„Hello“,TextToSpeech.QUEUE_FLUSH,null);
```

But when ?

Voice output

- Voice output technically not trivial
- Has to be initialized first
  - Can take a while
  - TextToSpeech gives a notification when it is ready
  - A `Listener` has to get that
  - the Activity has to implement the corresponding Interface

```
@override
 public void onInit(int status) {
    tts.speak("Hello",TextToSpeech.QUEUE_FLUSH,null);
    }
```

- The language can be defined with
  - `tts.setLanguage(Locale.GERMAN);`

Voice output

- The App now talks, but it says allways the same thing: „Hello“
- What can we do, so we can input some text, that the App then speaks
- We need
  - A text field for the Input
  - A button, that starts the voice output
- Another Listener needed?

What happens in the App so far

- At the start somewhere new MainActivity is executed
- In the onCreate method a new TextToSpeech object named tts is created
- This object send a notification when it is ready to speak, whereupon the onInit method is called automatically
- In onInit the output of the text is done with the call of tts.speak

Many things happen on the base of existing code written by other developers

the Manifest File

- Central description of the App

C:\Users\Rawiel\Documents\Vorlesungen\VG\AKG\AndroidApps\Projects\app\src\main\AndroidManif...

Datei Bearbeiten Suchen Ansicht Kodierung Sprachen Einstellungen Werkzeuge Makro Ausführen Erweiterungen
Fenster ?

AndroidManifest.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      package="de.akgaea.sagwaseinfach">
4
5      <application
6          android:allowBackup="true"
7          android:icon="@mipmap/ic_launcher"
8          android:label="@string/app_name"
9          android:roundIcon="@mipmap/ic_launcher_round"
10         android:supportsRtl="true"
11         android:theme="@style/Theme.SagWasEinfach">
12         <activity android:name=".MainActivity">
13             <intent-filter>
14                 <action android:name="android.intent.action.MAIN" />
15
16                 <category android:name="android.intent.category.LAUNCHER" />
17             </intent-filter>
18         </activity>
19     </application>
20
21 </manifest>
```

length : 748 lines : 21 Ln : 1 Col : 1 Pos : 1 Windows (CR LF) UTF-8 INS

Manifest-Datei

- Here Activities have to be registered
  - Possibly more than one
- Intent-Filter defines, which Activity is launch when the App is started
- Permissions
  - Right that possibly need to be granted to the App
    - Access to the camera
    - Acess to contacts
    - Access to location
    - etc.

```
<uses-permission android:name="android.permission.INTERNET"/>
```

Ressources

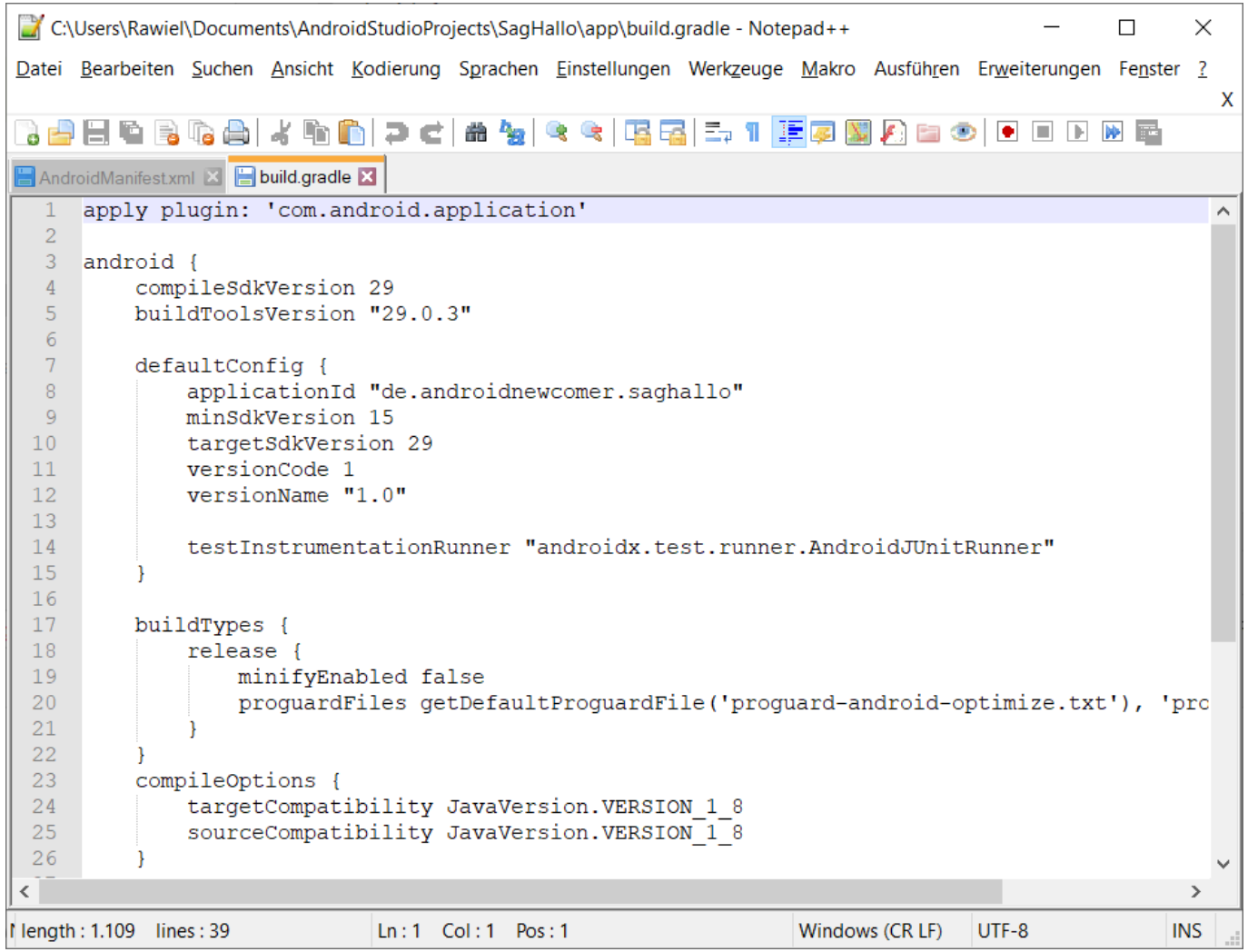
- Screen layouts
- Graphics
- Images
- Texts
- Icons

In the project tree under „res“

Build Scripts

- To call a Java Compiler is not enough
- Many tasks are necessaryto
  - Translate the code
  - Optimize the code
  - Sign it digitally
  - To put it all together with the ressources into an APK bundle
- That is what Build Tools take care of
- A Program named Gradle
  - Managed through script files named build.gradle

build.gradle



Graphical user interfaces

- activity_main.xml in the layout directory of the ressources
- Dobble click on the file opens the Layout editor
  - in the middle a graphical editor
  - palette with available Views, Layouts and Widgets
  - the attributes of the selected element

This Layout is shown due to the command

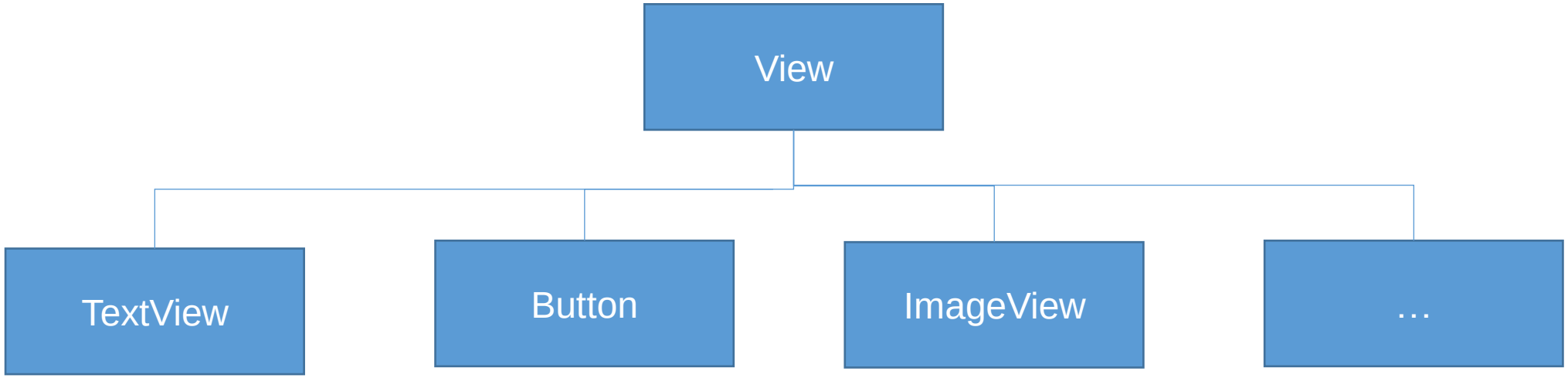
```
setContentView(R.layout.activity_main);
```

in onCreate()

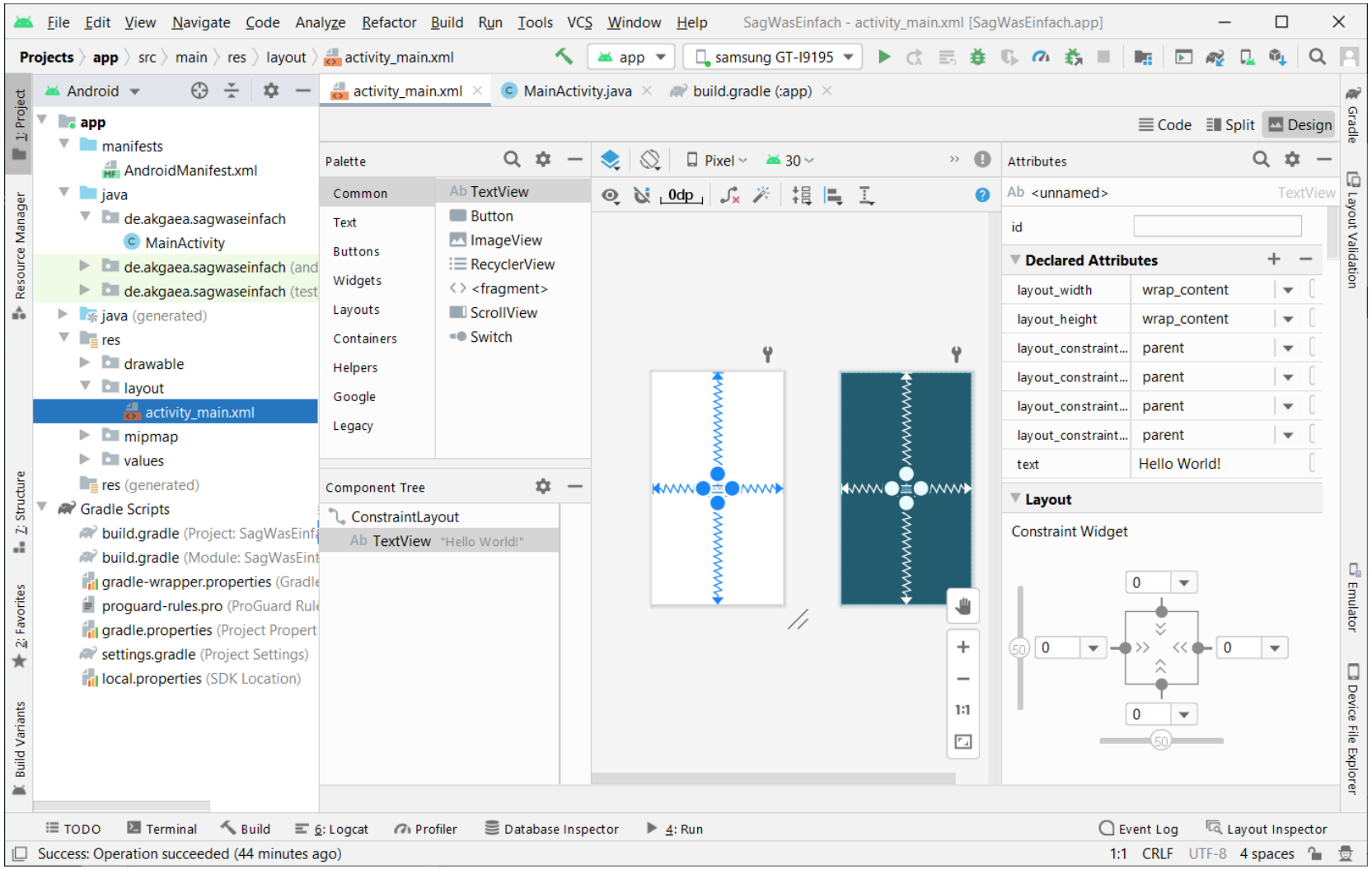
where R is a generated classthat cannot be edited

The Layout consists of Views

- Layouts are boxes
- Views are the content of boxes



Layout-Editor



Add Elements

- Add a button and a textfield for the input to the Layout out of the palette via drag & drop
- A look at the attributes of the button element
  - id – important for the identification of the button
  - text – for the labelling
  - additional attributes for the look
  - Attributes for the placement on the Layout
- important: every added element on the Layout Layout can be referenced over the class R and its Id in the java code

Buttons with Function

- How can funcionality be added to the button?
- The OnClickListener
- Where does that Listener have to be placed?
- What has to happen in the App?
  1. Initialize voice output
  2. Wait until the user presses the button
  3. Speak the text in the text field
- What is needed when?

Buttons with Function

- Waiting for the click works in the same way like waiting for the voice output to be redy
- where has the OnClickListener to be placed?
  - the OnClickListener has to be implemented by the MainActivity
- In the MainActivity an onClick method has to be implemented
- Shift the code for the voice output from the onInit-Methode to the onClick method
- Button has to use the running OnClickListener
  - `Button bSpeak = findViewByID(R.id.bspeak);`
  - `bSpeak.setOnClickListener(this);`
- → where does this code has to be placed?

Buttons with Function

- the onClick method
  - Is executed when the button is clicked
  - Because the button was connected to the Listener
- in the onClick method the text the user wrote into the text field shall be spoken
  - `EditText editText = (EditText)findViewById(R.id.txtSpeak);`
  - `tts.speak(editText.getText().toString(),TextToSpeeech.QUEUE_FLUSH, null, null)`
- → Test!