

MOBILE COMMERCE

Neema Mduma.

Applications Development with Android Studio

NM-AIST

Sep 01st, 2021



Learning Objectives

- Explain and apply principles of designing mobile applications while taking into account the limitations of mobile communication technologies.
- Know approaches of mobile web best practices and their effects on user acceptance.
- Know the basics of design principles / patterns / guidelines.
- Explain and apply basic properties of application development with Android.

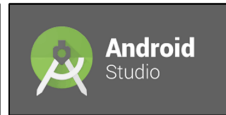


Android Basics

Android Development - Huge Offers



Dart

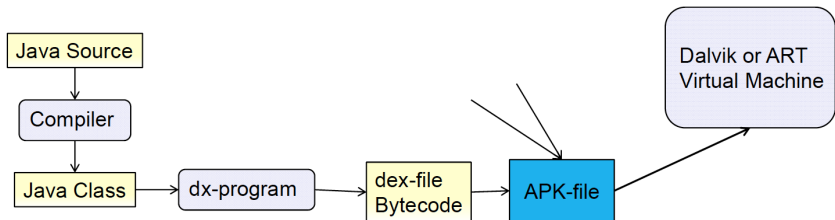




Java and Android (1/2)

Three main pillars:

- Programming language for Android development: Java (or Kotlin?)
- Standard class library (API) with sophisticated solutions for (almost) all routine tasks
- The runtime environment (Java Virtual Machine) with numerous tasks when executing Java programs





Java and Android (2/2)

Compared to the Java Standard Edition (JSE), Android-Java has a different standard library in order to serve to the special features of smartphones and tablets:

- Small displays with high variability in size and aspect ratio
- Touchscreen instead of a mouse
- Relatively low processor performance
- Applications that have moved into the background and no longer interact with the user can stop all resource-consuming activities
- Relatively small working memory: Applications running in the background are automatically shut down if there is insufficient memory
- Force to save energy
- Availability of special sensors (e.g. acceleration, GPS, camera)



Android Programming Environment

Android Studio (AS) consists of several parts that work together almost perfectly

- Android Studio IDE
- Android SDK Tools
- Android Operating System (actual Version 10.0)
- Android Emulator System with Google API

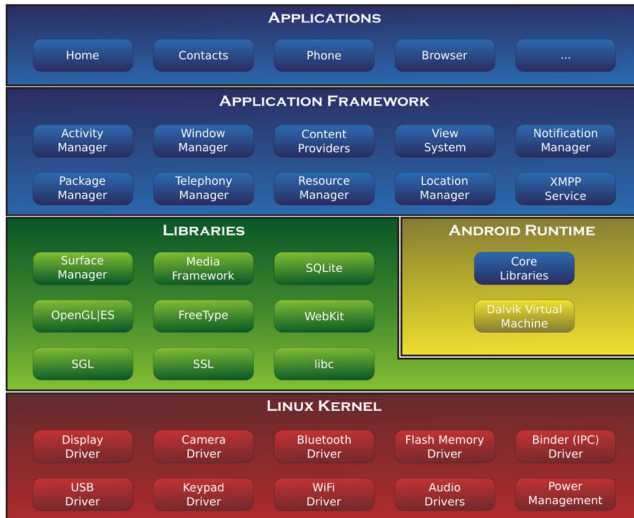
All important information, documentation, tutorials, guides, blogs and much more can be found under:

<https://developer.android.com>





Android System Architecture





Introduction to Android Components

Android is called a “Mobile Operating System”

- Modern platform for component-based applications
- Development of applications that access other applications (opening via authorizations), e.g. on pre-installed apps “Contacts”, “Phone”

The Main Components for Android Apps (realized of course through Java classes):

- Activity
- Service
- Content Provider
- Broadcast Receiver

Each App may consist of several components - at least one Activity



Most Important Component “Activity”

An activity presents a screen page with controls for a specific activity, e.g. viewing a list of contacts, editing details about a contact - interactive contact point to the App for the user.

- The GUI of an activity usually occupies the entire display area (Occasionally part of the display area). Handling of user interaction.
- The Main or Start activity appears when an app starts. A “kind” of main method.
- Apps usually comprise several activities between which the user can switch to perform certain tasks.
- The activities (in fact all components) of different Apps can communicate with each other - provided the permission is granted.



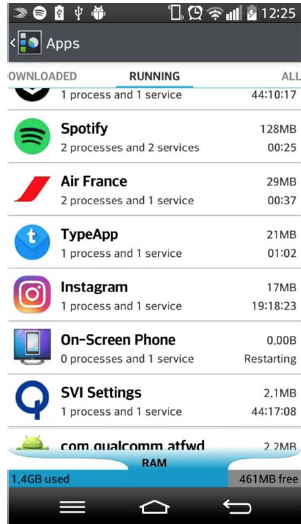
Component “Service” (1/2)

A service realizes a tasks in the background and has no user interface

- It can be started by another component to perform a single task in the background
- A service usually ends as soon as its task (e.g. a download, a synchronization, a music peace ...) is completed
- A started service continues even when the user changes to another activity. No further control by the initiator takes place after the start.
- A so called “bound service” can serve several clients and ends with the completion of the last binding.



Component “Service” (2/2)



The screenshot shows the 'Apps' settings screen on an Android device. The status bar at the top indicates the time is 12:25. The 'Apps' title bar is visible. Below it, there are three tabs: 'OWNLOADED', 'RUNNING' (which is selected), and 'ALL'. The 'RUNNING' tab shows a list of applications with their respective process and service counts and memory usage. At the bottom, there is a 'RAM' section showing '1.4GB used' and '461MB free'.

Application	Processes and Services	Memory Usage	Other Info
System	1 process and 1 service	44:10:17	
Spotify	2 processes and 2 services	128MB	00:25
Air France	2 processes and 1 service	29MB	00:37
TypeApp	1 process and 1 service	21MB	01:02
Instagram	1 process and 1 service	17MB	19:18:23
On-Screen Phone	0 processes and 1 service	0.00B	Restarting
SVI Settings	1 process and 1 service	2.1MB	44:17:08
com.qualcomm.atfwd		2.2MB	

RAM: 1.4GB used, 461MB free



Component “Content Provider”

Managing persistent data and controlling the access of other Apps

- Management of data and abstraction through an underlying persistence layer
- Provision of data on authorizations for other Apps
- Loose coupling to different Apps via a well defined interface
- Persistent data can be reside on any type of source: Files, SQL-database, Cloud, any URI-source
- and can be of any type



Component “Broadcast Receiver”

A broadcast receiver can react to broadcast messages that come from the system or from applications, e.g.

- Low battery status
- WLAN connection established
- Flight mode ended

No own user interface

May only be active for a short time

Can start activities or services and thus allow the application to be started on suitable occasion

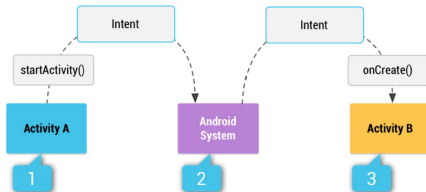


“Intent”: The Messaging Class between Components

They are used ...

- to place an order, e.g. “Start another Activity”
- to inform about an event (mostly at system level), e.g. WLAN activated

```
public void sendMessage (View view) {  
    Intent intent = new Intent(this, DisplayMessageActivity.class);  
    startActivity(intent);  
}
```



Intents are the
“channel-glue”
between
components



Transferring Data Between Activities

Intents can carry additional data of almost every type

- Standard types (int, double, boolean, String etc.)
- Classes that implements the interface parcelable (comparable to the standard java interface Serializable)
- The “extra”- data is appended as a [name, value] pair Start an Activity and send some data:

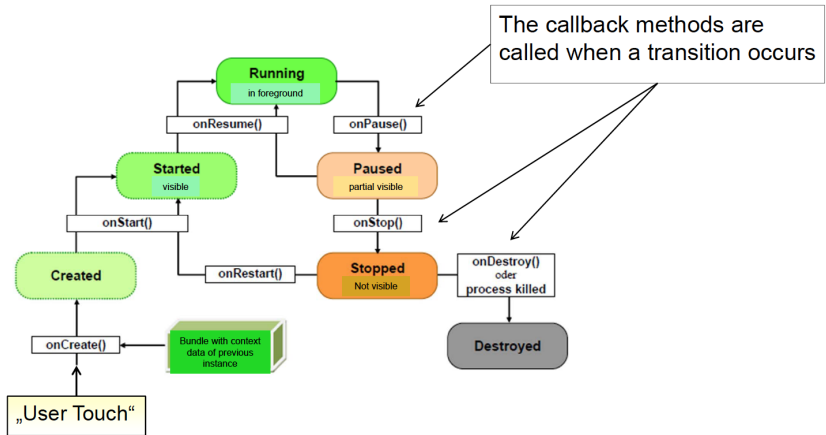
```
public void sendMessage(View view) {  
    Intent intent = new Intent(this, DisplayMessageActivity.class);  
    intent.putExtra("meskey", message);  
    startActivity(intent);  
}
```

Extract the received data:

```
Intent intent = getIntent();  
String s = intent.getStringExtra("meskey");
```



State Transition Diagram of an Activity





Android Studio System (1/2)

AS provides various wizards and helpful integrated tools:

- New project wizard: All initial setups for a new project and a simple runnable program frame will be done

Android SDK (Software Development Kit) Manager:

- Selection of different API levels, additional ones can be downloaded
- Projects can be rebuild and deployed for different API levels

Program Emulator:

- Select (or create) a virtual device to simulate a runtime hardware
- Connect your own real device via USB



Android Studio System (2/2)

Gradle is downloaded and set up as the build-tool:

- Taking care of software dependencies
- Checking and resolving references
- Building and packaging of Apps

Editor integrated UI design tool

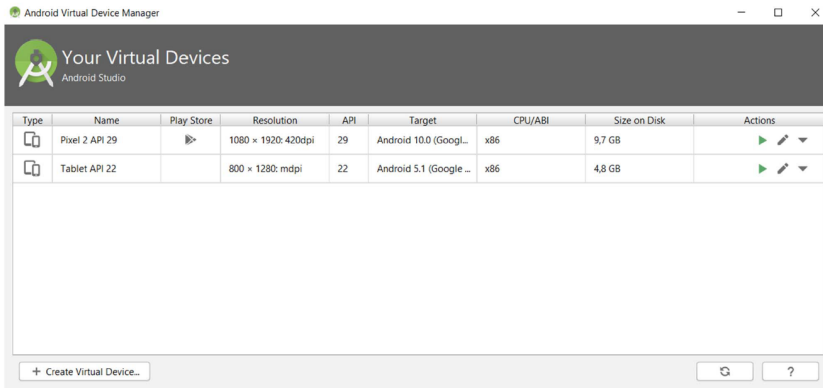
- WYSIWYG tool for building graphical user interfaces
- Generates an XML - representation of screen layouts

Debugger



Create your own “Virtual Smartphone”

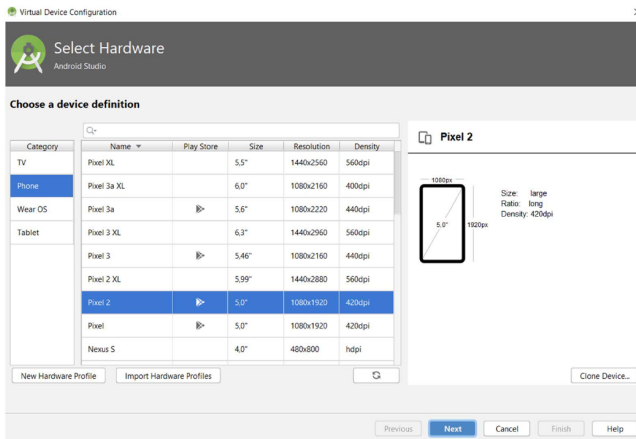
Tools : AVD - Manager (Android Virtual Device)





Create a Virtual Device

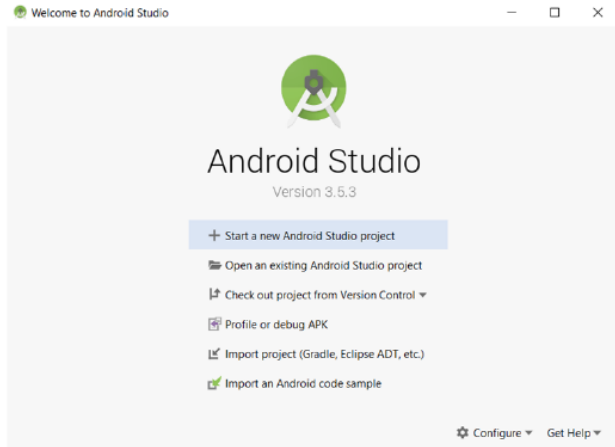
You can choose the offered device or “design” a complete other device (next slide)





Creating a New Project (1/3)

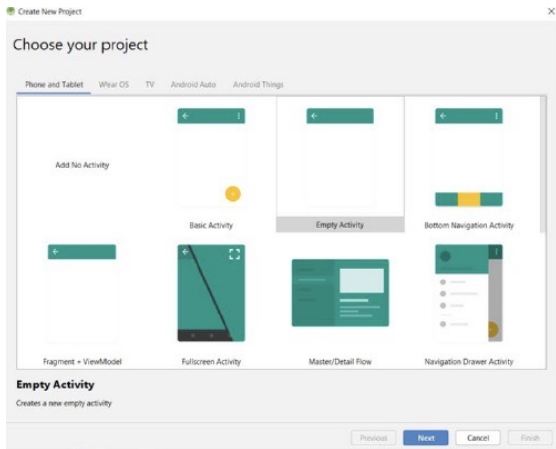
Select “Start a new Android Studio project”





Creating a New Project (2/3)

Some standard frames are predefined. We start with an Empty Activity





Creating a New Project (3/3)

Configure your project

←

Empty Activity

Creates a new empty activity

Name
HelloWorld

Package name
tz.ac.mp1st.neema.helloworld

Save location
c:\Project\HelloWorld

Language
Java

Minimum API level
API 29: Android 10.0 (Q)

i Your app will run on < 1% of devices.
[Help me choose](#)

☐ This project will support instant apps

☒ Use AndroidX artifacts

⚠ "HelloWorld" already exists at the specified project location.

Previous Next Cancel Finish

Name the title of the App

Package Name

Location of Project on Disk

Language = Java

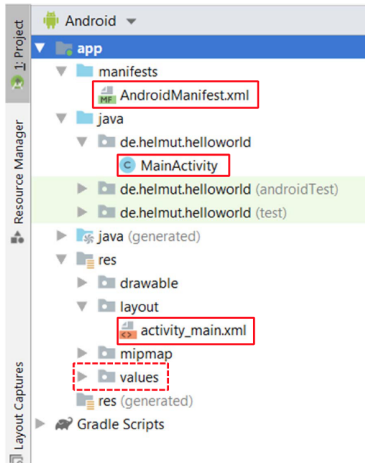
Minimum API-Level of target machines

After clicking finish the first project will be build (takes some time!)



Where to Find What in Android Studio ?

Directory structure of an (almost) empty Activity



The Manifest file: *AndroidManifest.xml*

MainActivity.java: The Java Start Activity class

GUI Layout for the *Main Activity*

Various Resources (later ...)



The Android Manifest (1/2)

The Manifest file is an XML - file containing metadata about an App

- Package name of the App, should be worldwide unique e.g.:
tz.ac.nm-aist.neema.helloworld (reversed domain name)
- Components belonging to an App
- Intents linking components: so called intent-filters
- Permissions

Good news: Usually this is well maintained by AS



The Android Manifest (2/2)

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
```

```
  package="tz.ac.nmaist.neema.helloworld">
```

Java package name

```
  <application
```

Application specification

```
    android:allowBackup="true"
```

```
    android:icon="@mipmap/ic_launcher"
```

```
    android:label="@string/app_name"
```

Reference to resource holding the name of the App

```
    android:roundIcon="@mipmap/ic_launcher_round"
```

```
    android:supportsRtl="true"
```

```
    android:theme="@style/AppTheme">
```

```
      <activity android:name=".MainActivity">
```

Class name of the Activity

```
        <intent-filter>
```

```
          <action android:name="android.intent.action.MAIN" />
```

```
          <category android:name="android.intent.category.LAUNCHER" />
```

```
        </intent-filter>
```

```
      </activity>
```

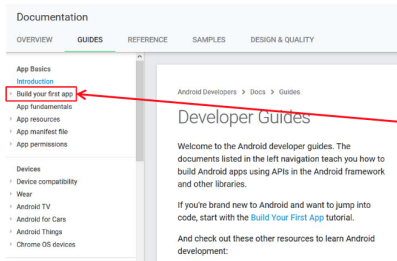
```
    </application>
```

This is a MAIN Activity
It can be LAUNCHED from the Start Menu



Android Guide: Build your first App

- The windows installer can be found under:
<https://developer.android.com/guide>
- Follow the guide to build your first App



- ▼ Build your first app
 - Overview
 - Create an Android project
 - Run your app
 - Build a simple user interface
 - Start another activity

Acknowledgment

Prof. Dr. Helmut Faasch, Stefan Wunderlich, Marius Wybrands,
Prof. Dr. Jorge Marx Gomez



VERY LARGE
BUSINESS APPLICATIONS