Pázmány Péter Catholic University
Faculty of Information Technology and Bionics

Android Development

Introduction, Basics

Kálmán Tornai
Room #232
tornai.kalman@itk.ppke.hu

Administration

- Class organization
  - Classes start at 10.15
  - There will be two holidays
    - April 8th – Easter holidays
    - April 15th – Easter holidays
  - Probably there will not be class on May 6th.
  - 3×45 minutes
    - mostly lecture with some coding

Foreword – Examination and evaluation system

- It is mandatory to participate in the lessons
  - The maximum number of allowed absence is 3
- On each lesson a short test-paper
  - Will be rated between 0 and 10.
  - Missing one is rated as 0.
- You must hand in your homework solutions.
  - You have to submit your homework solution before the next class starts
  - The solutions are rated, and the sum of the rate is calculated at the end of the semester.
  - Missing homework is 0 points
  - Solution without issues is 25 points

Foreword – Cont'd

- Project homework during the semester
  - More info will be given soon
  - 300 points
- Final grade
  - Points of short test + points of homeworks + points of project
  - Estimated maximal:100 + 200 + 300
  - Grading
    - [0% 50%): fail
    - [50% 60%): pass
    - [60% 70%): satisfactory
    - [70% 80%): good
    - [80% 100%]: excellent

Handing in the homework

- You will hand in the homework by using a repository
- It is mandatory to fill the following form ASAP
  - https://forms.gle/84wN8wBh64xdfMx88

Slack

- Invitation link
  - There is no need to register and join again, last semester's Slack workspace will be used.
  - However, you should join the new channel: #android2020

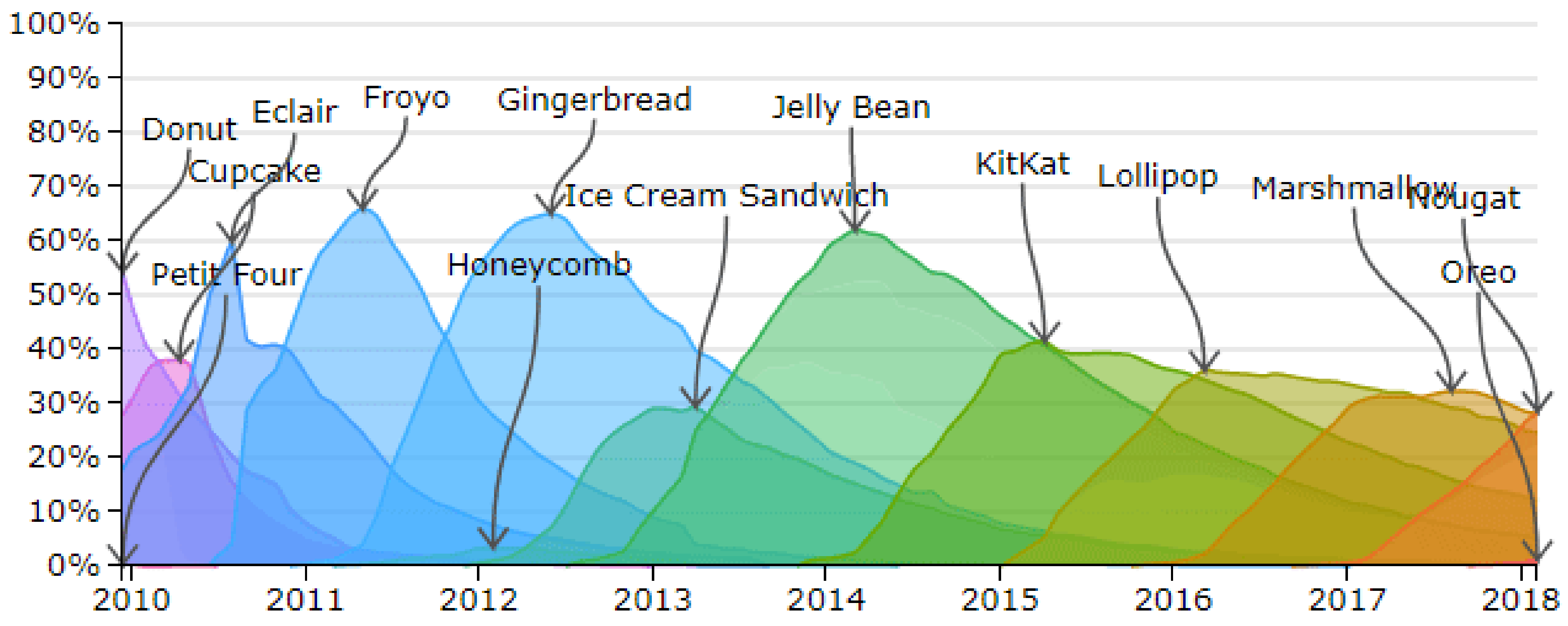
Android in a nutshell

Android properties

- Supervised and developed by Google – software package
  - Linux + Android VM + Other applications
  - Many manufacturer, different hardware
    - Mostly based on ARM, but x86 port is available, more and more Intel (MX5X) processor
    - Big variety in the hardware
    - Additional software which is not part of the base software
  - Development tools and emulator
    - Available for all platforms
- Main properties
  - Modular
  - Multitask, automatic memory managements, program libraries included
  - Almost arbitrary mobile communication technology is available (GSM … LTE)
  - Wi-Fi (Client and AP), Ethernet (tethering), Bluetooth, NFC
  - Sensors: GPS, Triaxial accelerometer / magnetometer, thermometer, light sensor
  - Camera support, recording and playback, even stereo
  - HDMI support, accelerated 2D and 3D graphics, parallel computation

Introduced in	Version number	Name	API LEVEL
2007	ß		ß
2008	1.0		1
2009	1.1		2
2009	1.5	Cupcake	3
2009	1.6	Donut	4
2009	2.0	Eclair	5
2010	2.2	Froyo	8
2010	2.3	Gingerbread	9
2011	3.0	Honeycomb	11
2011	4.0	Ice Cream Sandwich	14
2013	4.1	Jelly Bean	16
2013	4.4	KitKat	19
2014	5.0	Lollipop	21
2015	6.0	Marshmallow	23
2016	7.X	Nougat	24
2017	8.X	Oreo	26
2018	9.X	Pie	28
2019	10	Android 10	29

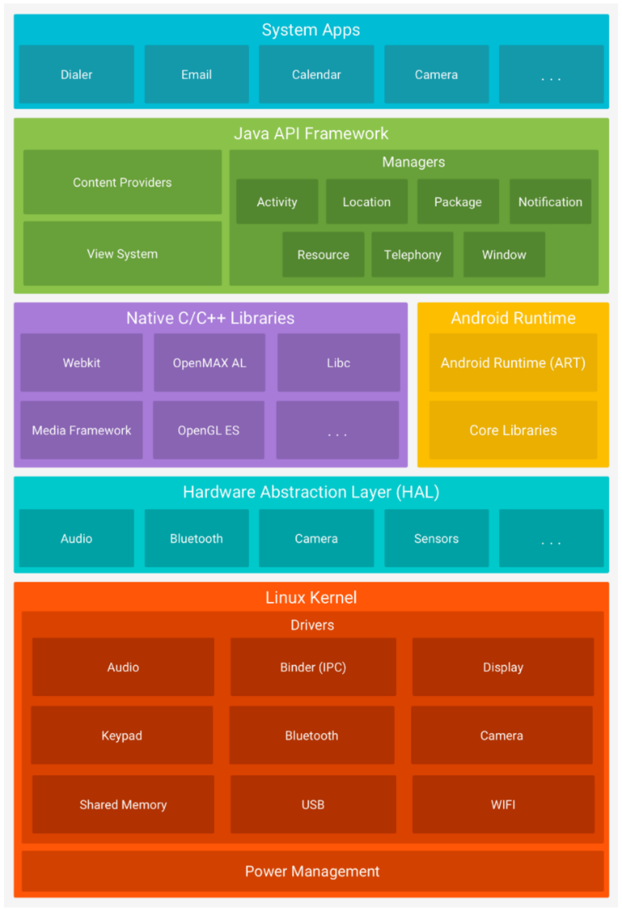
Spread of different versions



Android details

- Multiple software together
  - Middleware
  - Core applications
  - Operation system
- Java based programming language, but not Java!
  - Java packages missing from Android
    - java.applet
    - java.beans
    - javax.rmi
    - javax.print
  - Custom Virtual machine. Not the JVM, but ART(Android RunTime)
    - (Dalvik VM before Lollipop)
    - Open source
      - Takes less place
      - multiple VM can run simultaneously and better performance
    - Ahead-Of-Time compilation
      - Precompilation when the application is installed to the device
    - *.java → *.class → *.dex →*.apk

Building blocks of Android



Kotlin

- Kotlin – programming language
  - On Java virtual machine
  - First appeared in 2011
  - Last stable release: 1.3.61 – November 2019.
  - For Android since 2017 Google I/O
  - Kotlin is designed to be an industrial-strength object-oriented language, and a "better language" than Java, but still be fully interoperable with Java code

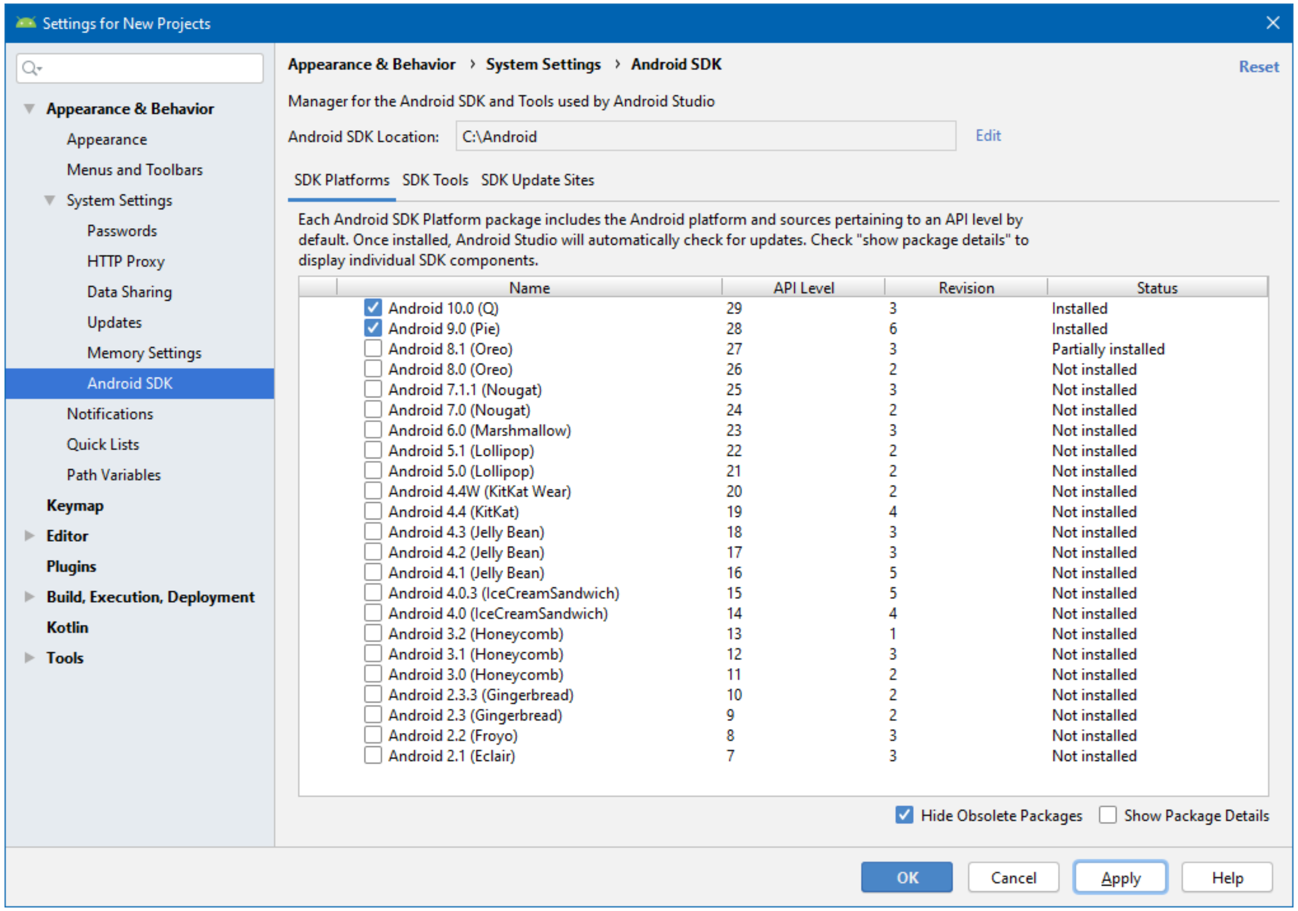
Android app components

- Building blocks
  - Activity
    - Basically „windows"
    - Every graphical user interface (GUI) in the app belongs to an Activity
    - Best practice is to package the same functionality within one activity
      - For example login screen, route planning, etc. – we will see more from this later.
  - Service
    - Tasks in the background without UI
    - For example, playing music, sync with server etc.
    - Can run forward in the background after the user opens an another app
  - Broadcast Receiver
    - Trigger specific tasks for specific events
  - Content Provider
    - These parts of the app can arrange the data stored on the device
    - Make it accessible for one or multiple applications in an easy way

Android developer tools

- Android Studio
  - We (and basically everyone) use this IDE
  - Based on IntelliJ + additions for android development
  - Free software, the newest is 3.5
- Android SDK
  - Compiler and software library
  - Emulator
    - It is possible to run an emulator on an another operating system (Windows), inside which the Android runs
    - Possible to test your applications on a custom „phone"
    - We will use real phones
- Android NDK
  - Native (C++) development is also possible

Recommended SDK settings



Appearance & Behavior › System Settings › Android SDK

Manager for the Android SDK and Tools used by Android Studio

Android SDK Location: C:\Android

Each Android SDK Platform package includes the Android platform and sources pertaining to an API level by default. Once installed, Android Studio will automatically check for updates. Check "show package details" to display individual SDK components.

	Name	API Level	Revision	Status
✓	Android 10.0 (Q)	29	3	Installed
✓	Android 9.0 (Pie)	28	6	Installed
☐	Android 8.1 (Oreo)	27	3	Partially installed
☐	Android 8.0 (Oreo)	26	2	Not installed
☐	Android 7.1.1 (Nougat)	25	3	Not installed
☐	Android 7.0 (Nougat)	24	2	Not installed
☐	Android 6.0 (Marshmallow)	23	3	Not installed
☐	Android 5.1 (Lollipop)	22	2	Not installed
☐	Android 5.0 (Lollipop)	21	2	Not installed
☐	Android 4.4W (KitKat Wear)	20	2	Not installed
☐	Android 4.4 (KitKat)	19	4	Not installed
☐	Android 4.3 (Jelly Bean)	18	3	Not installed
☐	Android 4.2 (Jelly Bean)	17	3	Not installed
☐	Android 4.1 (Jelly Bean)	16	5	Not installed
☐	Android 4.0.3 (IceCreamSandwich)	15	5	Not installed
☐	Android 4.0 (IceCreamSandwich)	14	4	Not installed
☐	Android 3.2 (Honeycomb)	13	1	Not installed
☐	Android 3.1 (Honeycomb)	12	3	Not installed
☐	Android 3.0 (Honeycomb)	11	2	Not installed
☐	Android 2.3.3 (Gingerbread)	10	2	Not installed
☐	Android 2.3 (Gingerbread)	9	2	Not installed
☐	Android 2.2 (Froyo)	8	3	Not installed
☐	Android 2.1 (Eclair)	7	3	Not installed

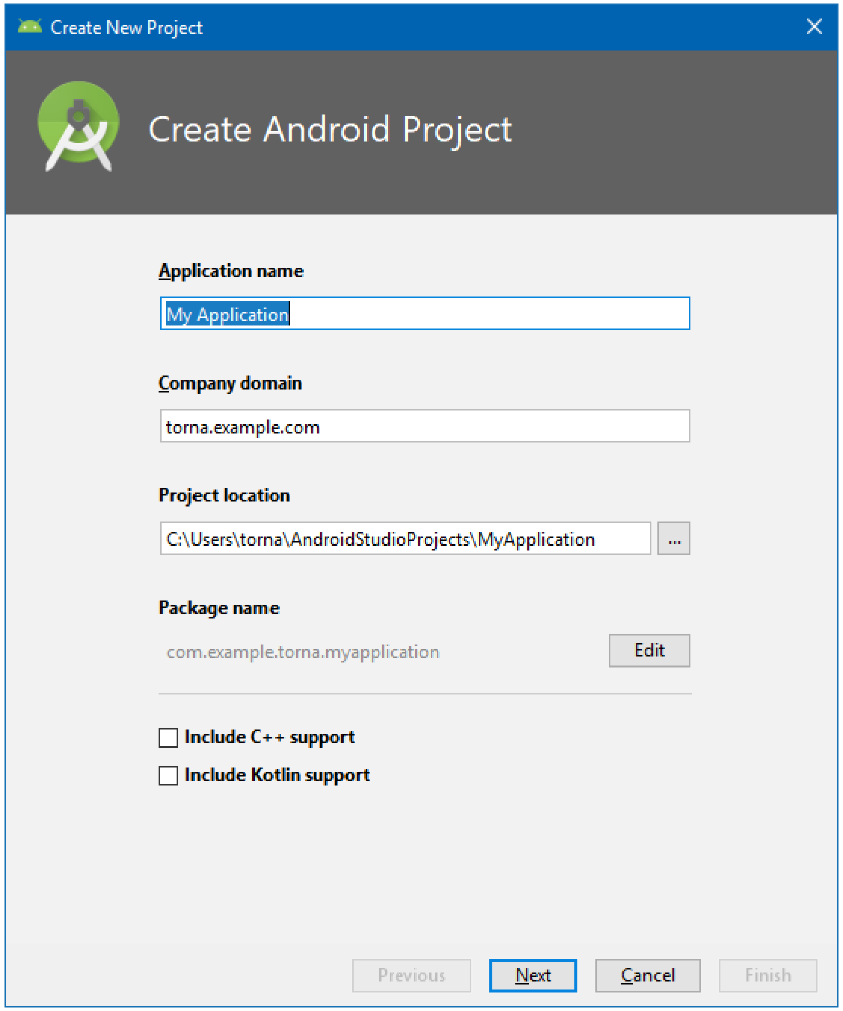
✓ Hide Obsolete Packages ☐ Show Package Details

Setup

- You should install (some of them are already installed)
  - „Tools” folder
    - Android SDK Tools
    - Android SDK Platform-tools
    - Android SDK Build-tools
  - „Android O” (10.0) folder (older versions are also supported)
    - SDK Platform
    - Intel x86 Atom System Image
    - Sources for Android SDK
  - „Extras” folder
    - Android Support Library
    - Google Play Services
    - Google USB Driver (Windows)
    - Intel x86 Emulator Accelerator (HAXM Installer) – if you have appropriate Intel CPU

Hello Android

- Let’s create a new Android app
- On welcome screen choose „Start new Android Studio project”
- Set the name of the project
- Set the company domain
    - Package name is generated
    - Package name should be unique
        - mad.itk.ppke.hu


Create New Project

Create Android Project

Application name: My Application

Company domain: torna.example.com

Project location: C:\Users\torna\AndroidStudioProjects\MyApplication

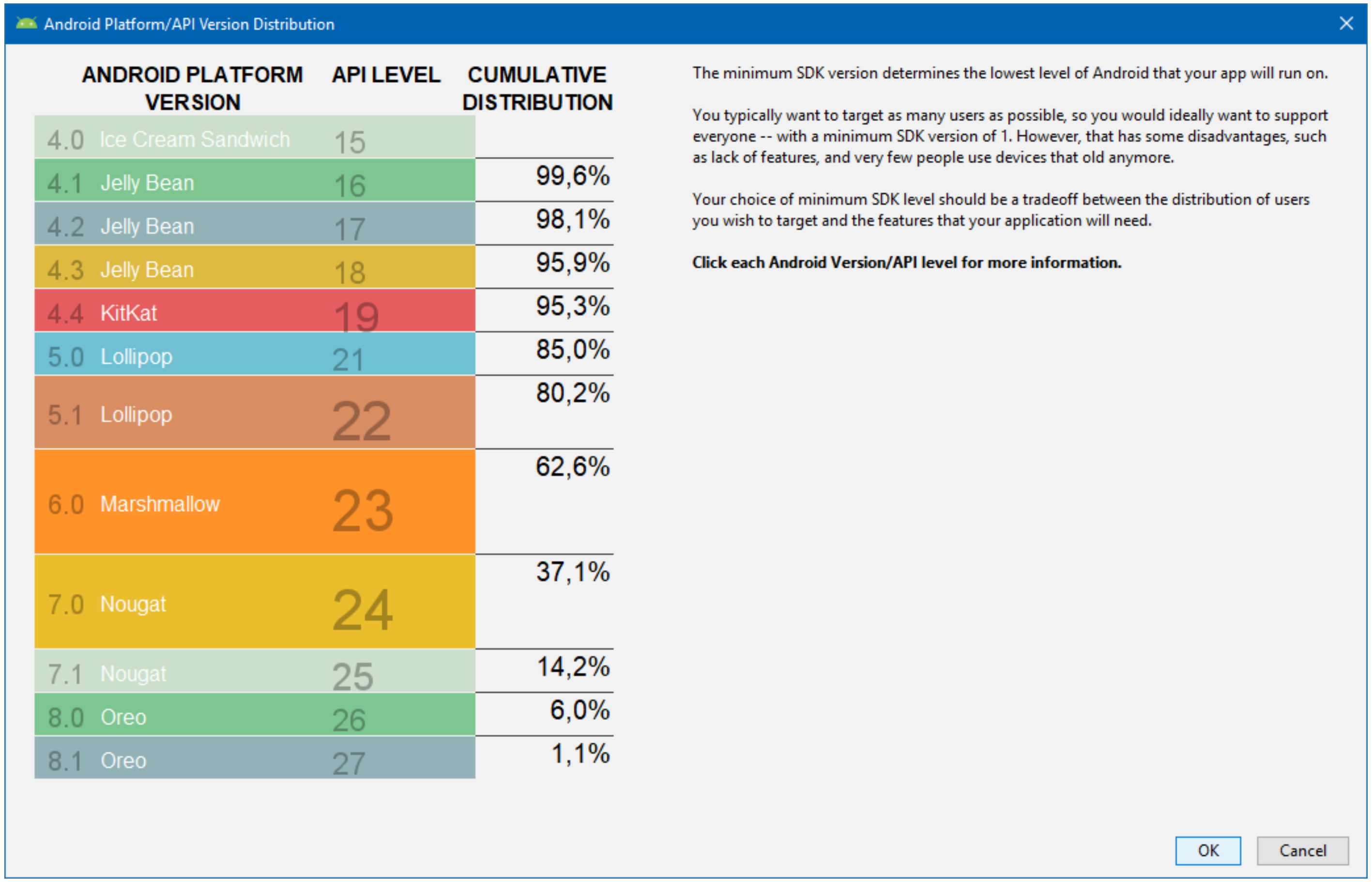
Package name: com.example.torna.myapplication

☐ Include C++ support
☐ Include Kotlin support

Hello Android

- Minimum SDK version: the oldest Android version, which is supported by the app
  - If it is too low, many of new API components cannot be used
  - If it is too high, only a few device will be supported
- Target SDK version: which capabilities wanted to be utilized
  - You should choose the latest one
- Compile with: which used for compilation
  - You should choose the latest one as well

Hello Android



Android Platform/API Version Distribution

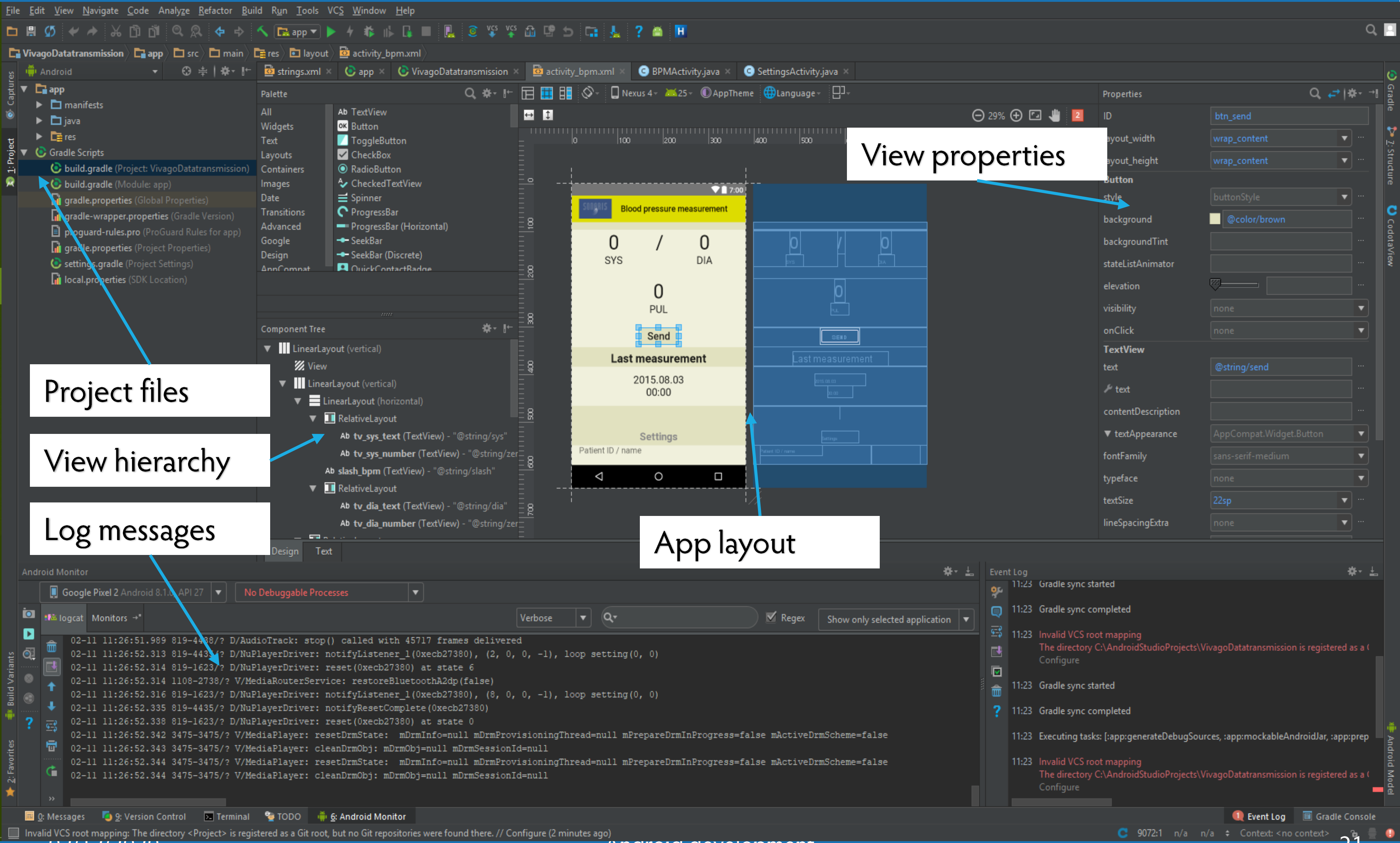
ANDROID PLATFORM VERSION		API LEVEL	CUMULATIVE DISTRIBUTION
4.0	Ice Cream Sandwich	15	
4.1	Jelly Bean	16	99,6%
4.2	Jelly Bean	17	98,1%
4.3	Jelly Bean	18	95,9%
4.4	KitKat	19	95,3%
5.0	Lollipop	21	85,0%
5.1	Lollipop	22	80,2%
6.0	Marshmallow	23	62,6%
7.0	Nougat	24	37,1%
7.1	Nougat	25	14,2%
8.0	Oreo	26	6,0%
8.1	Oreo	27	1,1%

The minimum SDK version determines the lowest level of Android that your app will run on.

You typically want to target as many users as possible, so you would ideally want to support everyone -- with a minimum SDK version of 1. However, that has some disadvantages, such as lack of features, and very few people use devices that old anymore.

Your choice of minimum SDK level should be a tradeoff between the distribution of users you wish to target and the features that your application will need.

Click each Android Version/API level for more information.



Properties:

ID	btn_send
layout_width	wrap_content
layout_height	wrap_content
style	buttonStyle
background	@color/brown
visibility	none
onClick	none
text	@string/send
textAppearance	AppCompat.Widget.Button
fontFamily	sans-serif-medium
typeface	none
textSize	22sp
lineSpacingExtra	none

Component Tree:

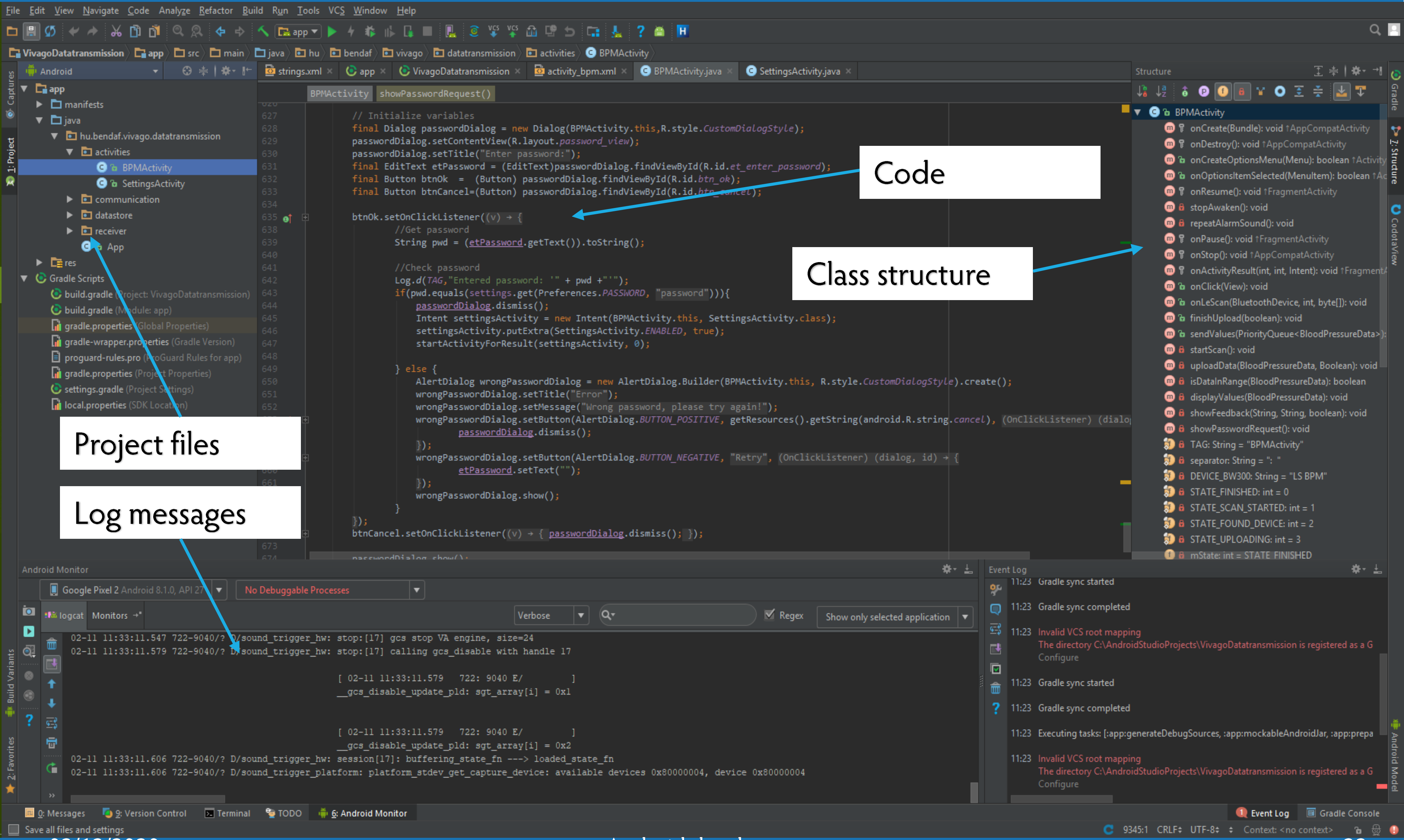
```
LinearLayout (vertical)
  View
  LinearLayout (vertical)
    LinearLayout (horizontal)
      RelativeLayout
        tv_sys_text (TextView) - "@string/sys"
        tv_sys_number (TextView) - "@string/zer
      slash_bpm (TextView) - "@string/slash"
      RelativeLayout
        tv_dia_text (TextView) - "@string/dia"
        tv_dia_number (TextView) - "@string/zer
```

```
02-11 11:26:51.989 819-4438/? D/AudioTrack: stop() called with 45717 frames delivered
02-11 11:26:52.313 819-4435/? D/NuPlayerDriver: notifyListener_l(0xecb27380), (2, 0, 0, -1), loop setting(0, 0)
02-11 11:26:52.314 819-1623/? D/NuPlayerDriver: reset(0xecb27380) at state 6
02-11 11:26:52.314 1108-2738/? V/MediaRouterService: restoreBluetoothA2dp(false)
02-11 11:26:52.316 819-1623/? D/NuPlayerDriver: notifyListener_l(0xecb27380), (8, 0, 0, -1), loop setting(0, 0)
02-11 11:26:52.335 819-4435/? D/NuPlayerDriver: notifyResetComplete(0xecb27380)
02-11 11:26:52.338 819-1623/? D/NuPlayerDriver: reset(0xecb27380) at state 0
02-11 11:26:52.342 3475-3475/? V/MediaPlayer: resetDrmState:  mDrmInfo=null mDrmProvisioningThread=null mPrepareDrmInProgress=false mActiveDrmScheme=false
02-11 11:26:52.343 3475-3475/? V/MediaPlayer: cleanDrmObj: mDrmObj=null mDrmSessionId=null
02-11 11:26:52.344 3475-3475/? V/MediaPlayer: resetDrmState:  mDrmInfo=null mDrmProvisioningThread=null mPrepareDrmInProgress=false mActiveDrmScheme=false
02-11 11:26:52.344 3475-3475/? V/MediaPlayer: cleanDrmObj: mDrmObj=null mDrmSessionId=null
```

Event Log:

```
11:23  Gradle sync started
11:23  Gradle sync completed
11:23  Invalid VCS root mapping
       The directory C:\AndroidStudioProjects\VivagoDatatransmission is registered as a
       Configure
11:23  Gradle sync started
11:23  Gradle sync completed
11:23  Executing tasks: [:app:generateDebugSources, :app:mockableAndroidJar, :app:prep
11:23  Invalid VCS root mapping
       The directory C:\AndroidStudioProjects\VivagoDatatransmission is registered as a
       Configure
```

Invalid VCS root mapping: The directory <Project> is registered as a Git root, but no Git repositories were found there. // Configure (2 minutes ago)

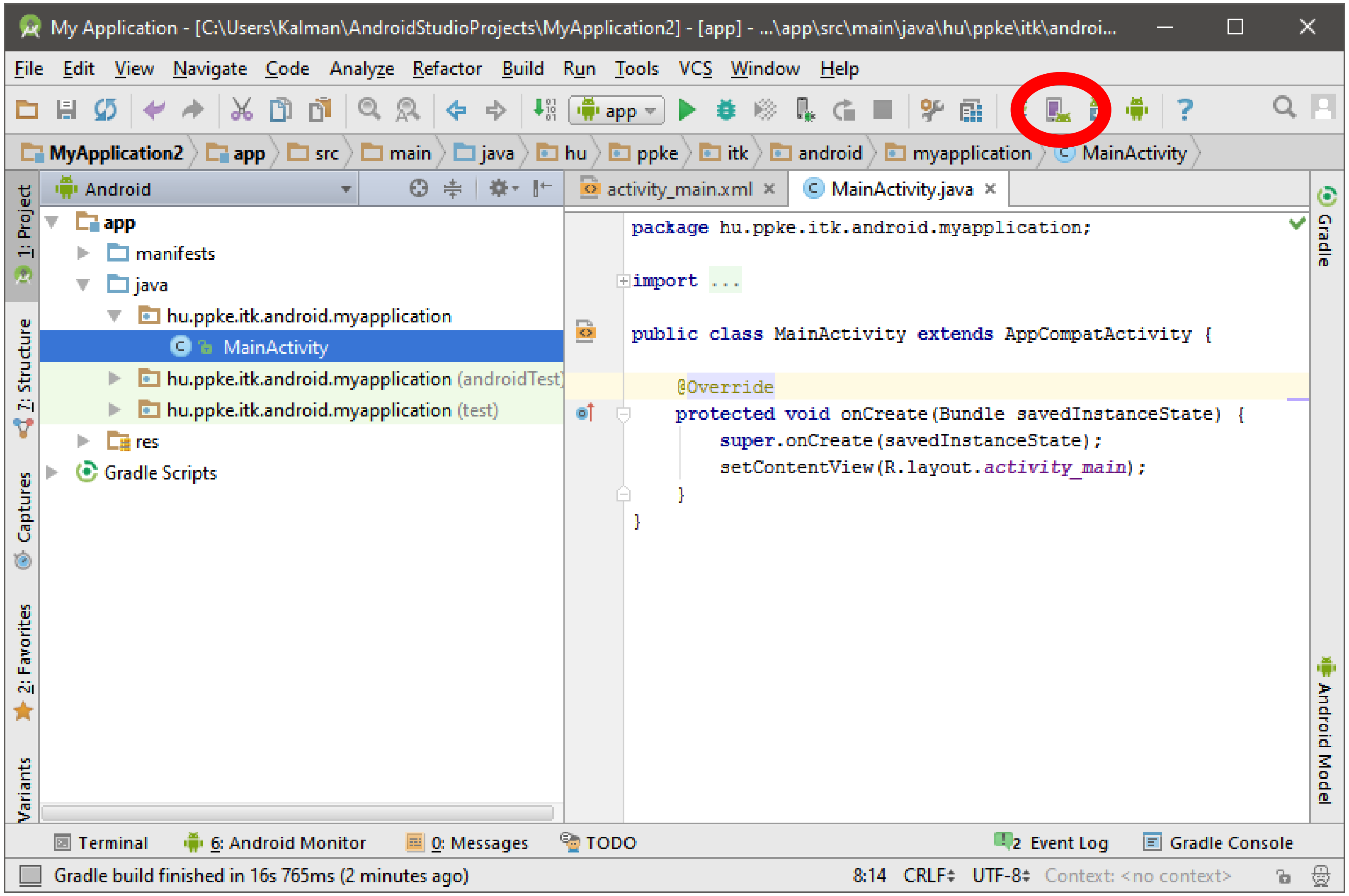

Code
Class structure
Project files
Log messages
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
VivagoDatatransmission app src main java hu bendaf vivago datatransmission activities BPMActivity
strings.xml app VivagoDatatransmission activity_bpm.xml BPMActivity.java SettingsActivity.java
BPMActivity showPasswordRequest()
Android
app
manifests
java
hu.bendaf.vivago.datatransmission
activities
BPMActivity
SettingsActivity
communication
datastore
receiver
App
res
Gradle Scripts
build.gradle (Project: VivagoDatatransmission)
build.gradle (Module: app)
gradle.properties (Global Properties)
gradle-wrapper.properties (Gradle Version)
proguard-rules.pro (ProGuard Rules for app)
gradle.properties (Project Properties)
settings.gradle (Project Settings)
local.properties (SDK Location)
Structure
Android Monitor
Google Pixel 2 Android 8.1.0, API 27
No Debuggable Processes
logcat
Monitors
Verbose
Regex
Show only selected application
Event Log
0: Messages
9: Version Control
Terminal
TODO
6: Android Monitor
Event Log
Gradle Console
Save all files and settings
9345:1 CRLF UTF-8 Context: <no context>

```java
        // Initialize variables
        final Dialog passwordDialog = new Dialog(BPMActivity.this,R.style.CustomDialogStyle);
        passwordDialog.setContentView(R.layout.password_view);
        passwordDialog.setTitle("Enter password:");
        final EditText etPassword = (EditText)passwordDialog.findViewById(R.id.et_enter_password);
        final Button btnOk  =  (Button) passwordDialog.findViewById(R.id.btn_ok);
        final Button btnCancel=(Button) passwordDialog.findViewById(R.id.btn_cancel);

        btnOk.setOnClickListener((v) → {
                //Get password
                String pwd = (etPassword.getText()).toString();

                //Check password
                Log.d(TAG,"Entered password: '" + pwd +"'");
                if(pwd.equals(settings.get(Preferences.PASSWORD, "password"))){
                    passwordDialog.dismiss();
                    Intent settingsActivity = new Intent(BPMActivity.this, SettingsActivity.class);
                    settingsActivity.putExtra(SettingsActivity.ENABLED, true);
                    startActivityForResult(settingsActivity, 0);

                } else {
                    AlertDialog wrongPasswordDialog = new AlertDialog.Builder(BPMActivity.this, R.style.CustomDialogStyle).create();
                    wrongPasswordDialog.setTitle("Error");
                    wrongPasswordDialog.setMessage("Wrong password, please try again!");
                    wrongPasswordDialog.setButton(AlertDialog.BUTTON_POSITIVE, getResources().getString(android.R.string.cancel), (OnClickListener) (dialo
                            passwordDialog.dismiss();
                    });
                    wrongPasswordDialog.setButton(AlertDialog.BUTTON_NEGATIVE, "Retry", (OnClickListener) (dialog, id) → {
                            etPassword.setText("");
                    });
                    wrongPasswordDialog.show();
                }
        });
        btnCancel.setOnClickListener((v) → { passwordDialog.dismiss(); });
```

```
BPMActivity
    onCreate(Bundle): void ↑AppCompatActivity
    onDestroy(): void ↑AppCompatActivity
    onCreateOptionsMenu(Menu): boolean ↑Activity
    onOptionsItemSelected(MenuItem): boolean ↑Ac
    onResume(): void ↑FragmentActivity
    stopAwaken(): void
    repeatAlarmSound(): void
    onPause(): void ↑FragmentActivity
    onStop(): void ↑AppCompatActivity
    onActivityResult(int, int, Intent): void ↑FragmentA
    onClick(View): void
    onLeScan(BluetoothDevice, int, byte[]): void
    finishUpload(boolean): void
    sendValues(PriorityQueue<BloodPressureData>):
    startScan(): void
    uploadData(BloodPressureData, Boolean): void
    isDataInRange(BloodPressureData): boolean
    displayValues(BloodPressureData): void
    showFeedback(String, String, boolean): void
    showPasswordRequest(): void
    TAG: String = "BPMActivity"
    separator: String = ": "
    DEVICE_BW300: String = "LS BPM"
    STATE_FINISHED: int = 0
    STATE_SCAN_STARTED: int = 1
    STATE_FOUND_DEVICE: int = 2
    STATE_UPLOADING: int = 3
    mState: int = STATE_FINISHED
```

```
02-11 11:33:11.547 722-9040/? D/sound_trigger_hw: stop:[17] gcs stop VA engine, size=24
02-11 11:33:11.579 722-9040/? D/sound_trigger_hw: stop:[17] calling gcs_disable with handle 17

                                                   [ 02-11 11:33:11.579   722: 9040 E/         ]
                                                   __gcs_disable_update_pld: sgt_array[i] = 0x1


                                                   [ 02-11 11:33:11.579   722: 9040 E/         ]
                                                   __gcs_disable_update_pld: sgt_array[i] = 0x2
02-11 11:33:11.606 722-9040/? D/sound_trigger_hw: session[17]: buffering_state_fn ---> loaded_state_fn
02-11 11:33:11.606 722-9040/? D/sound_trigger_platform: platform_stdev_get_capture_device: available devices 0x80000004, device 0x80000004
```

```
11:23 Gradle sync started

11:23 Gradle sync completed

11:23 Invalid VCS root mapping
      The directory C:\AndroidStudioProjects\VivagoDatatransmission is registered as a G
      Configure

11:23 Gradle sync started

11:23 Gradle sync completed

11:23 Executing tasks: [:app:generateDebugSources, :app:mockableAndroidJar, :app:prepa

11:23 Invalid VCS root mapping
      The directory C:\AndroidStudioProjects\VivagoDatatransmission is registered as a G
      Configure
```

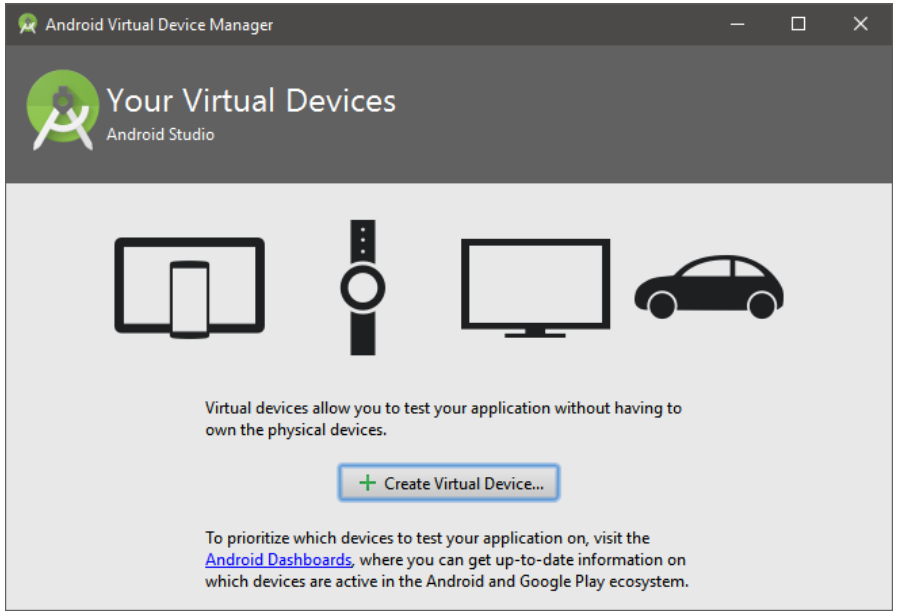

My Application - [C:\Users\Kalman\AndroidStudioProjects\MyApplication2] - [app] - ...\app\src\main\java\hu\ppke\itk\androi...
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MyApplication2 app src main java hu ppke itk android myapplication MainActivity
Android
app
manifests
java
hu.ppke.itk.android.myapplication
MainActivity
hu.ppke.itk.android.myapplication (androidTest)
hu.ppke.itk.android.myapplication (test)
res
Gradle Scripts
activity_main.xml
MainActivity.java
Terminal
6: Android Monitor
0: Messages
TODO
Event Log
Gradle Console
8:14 CRLF UTF-8 Context: <no context>

```
package hu.ppke.itk.android.myapplication;

import ...

public class MainActivity extends AppCompatActivity {

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

Gradle build finished in 16s 765ms (2 minutes ago)

Setting an emulator

- AVD Manger
  - A virtual device can be
    - created
    - deleted
    - started
    - modified
- Each device has an „disk" image, which is used by the emulator
  - Thus we have persistent storage

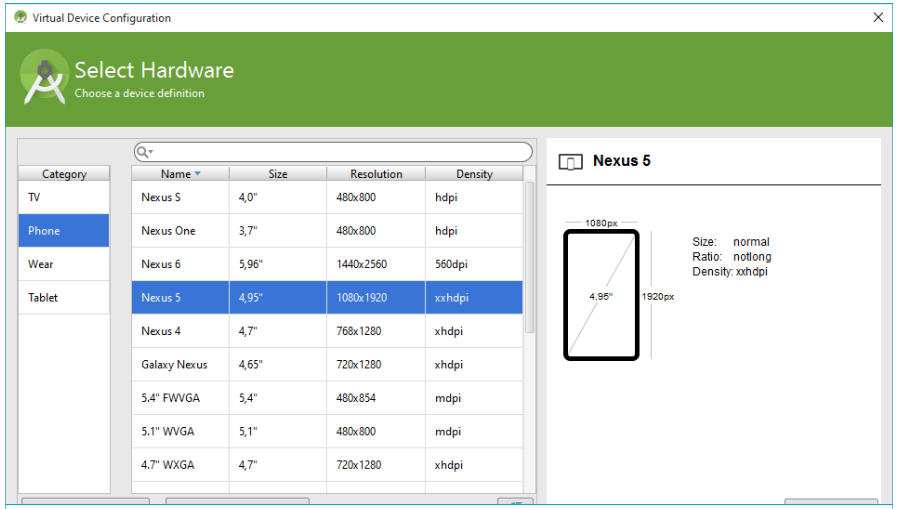


Virtual devices allow you to test your application without having to own the physical devices.

To prioritize which devices to test your application on, visit the Android Dashboards, where you can get up-to-date information on which devices are active in the Android and Google Play ecosystem.

Setting an emulator

- Create a new one
  - Device: Pixel
  - Set the previously downloaded Android version
    - 10.0
    - Architecture x86
    - With Google API


Virtual Device Configuration

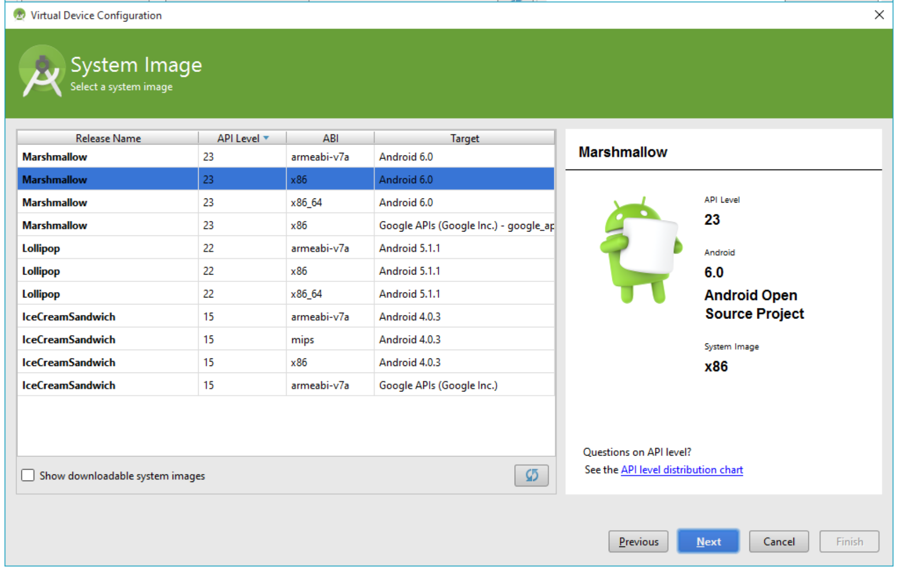
Select Hardware
Choose a device definition

Category
TV
Phone
Wear
Tablet

Name	Size	Resolution	Density
Nexus S	4,0"	480x800	hdpi
Nexus One	3,7"	480x800	hdpi
Nexus 6	5,96"	1440x2560	560dpi
Nexus 5	4,95"	1080x1920	xxhdpi
Nexus 4	4,7"	768x1280	xhdpi
Galaxy Nexus	4,65"	720x1280	xhdpi
5.4" FWVGA	5,4"	480x854	mdpi
5.1" WVGA	5,1"	480x800	mdpi
4.7" WXGA	4,7"	720x1280	xhdpi

Nexus 5

Size: normal
Ratio: notlong
Density: xxhdpi


Virtual Device Configuration

System Image
Select a system image

Release Name	API Level	ABI	Target
Marshmallow	23	armeabi-v7a	Android 6.0
Marshmallow	23	x86	Android 6.0
Marshmallow	23	x86_64	Android 6.0
Marshmallow	23	x86	Google APIs (Google Inc.) - google_ap
Lollipop	22	armeabi-v7a	Android 5.1.1
Lollipop	22	x86	Android 5.1.1
Lollipop	22	x86_64	Android 5.1.1
IceCreamSandwich	15	armeabi-v7a	Android 4.0.3
IceCreamSandwich	15	mips	Android 4.0.3
IceCreamSandwich	15	x86	Android 4.0.3
IceCreamSandwich	15	armeabi-v7a	Google APIs (Google Inc.)

☐ Show downloadable system images

Marshmallow

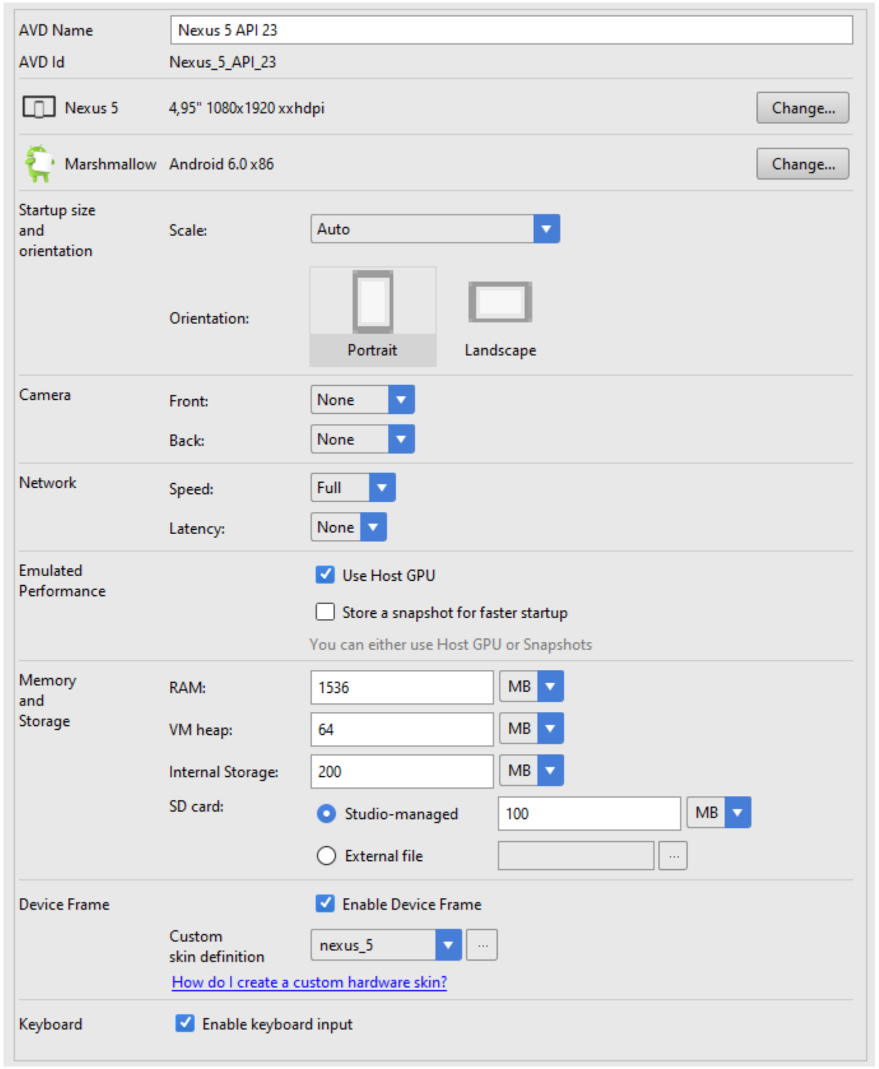
API Level
23

Android
6.0
Android Open Source Project

System Image
x86

Questions on API level?
See the API level distribution chart

Advanced settings



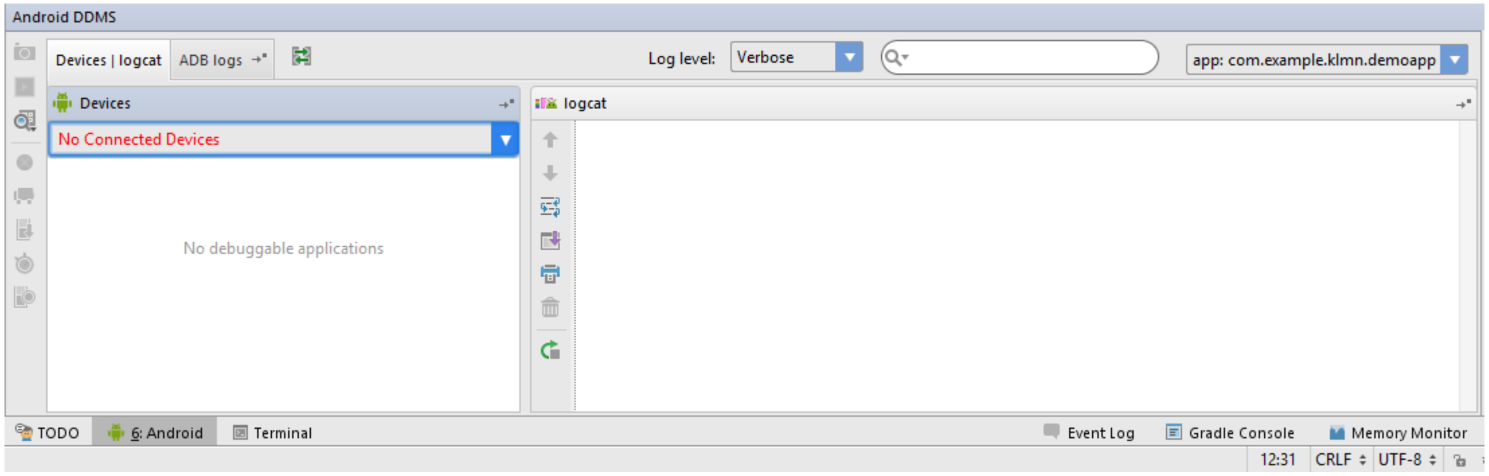
AVD Name	Nexus 5 API 23	
AVD Id	Nexus_5_API_23	
Nexus 5	4,95" 1080x1920 xxhdpi	Change...
Marshmallow	Android 6.0 x86	Change...
Startup size and orientation	Scale:	Auto
	Orientation:	[✓ Portrait] Landscape
Camera	Front:	None
	Back:	None
Network	Speed:	Full
	Latency:	None
Emulated Performance		✓ Use Host GPU
		Store a snapshot for faster startup
		You can either use Host GPU or Snapshots
Memory and Storage	RAM:	1536 MB
	VM heap:	64 MB
	Internal Storage:	200 MB
	SD card:	● Studio-managed 100 MB
		○ External file
Device Frame		✓ Enable Device Frame
	Custom skin definition	nexus_5
	How do I create a custom hardware skin?	
Keyboard	✓ Enable keyboard input	

Android Debug Bridge (ADB)

- Tool which communicates between the device and the IDE (or console)
- Client-server program:
  - Client
    - Runs on the development machine, command line application
  - Daemon thread
    - Runs in the background on each emulator or real device
  - Server
    - Background process, also runs on the development machine
    - Manages the communication between the client and the adb daemon
  - Server listens on the 5037 TCP port
    - Always communicate here with the client
  - Server are connecting with each device on a separate port between 5555-5585
  - Every device uses two ports:
    - Even for console connection, odd for adb connection
    - Emulator 1, console: 5554
      Emulator 1, adb: 5555
      Emulator 2, console: 5556
      Emulator 2, adb: 5557 ...

LogCat

- Informative messages can be sent to the console of the PC
- Use the static functions of the `android.util.Log` class
  - `Log.i("MainActivity", "Hello logging!"); // information log`
  - First parameter: label – you may want to write the classname here
  - Second parameter: message
- In Android Studio press Alt + 6 to open the console

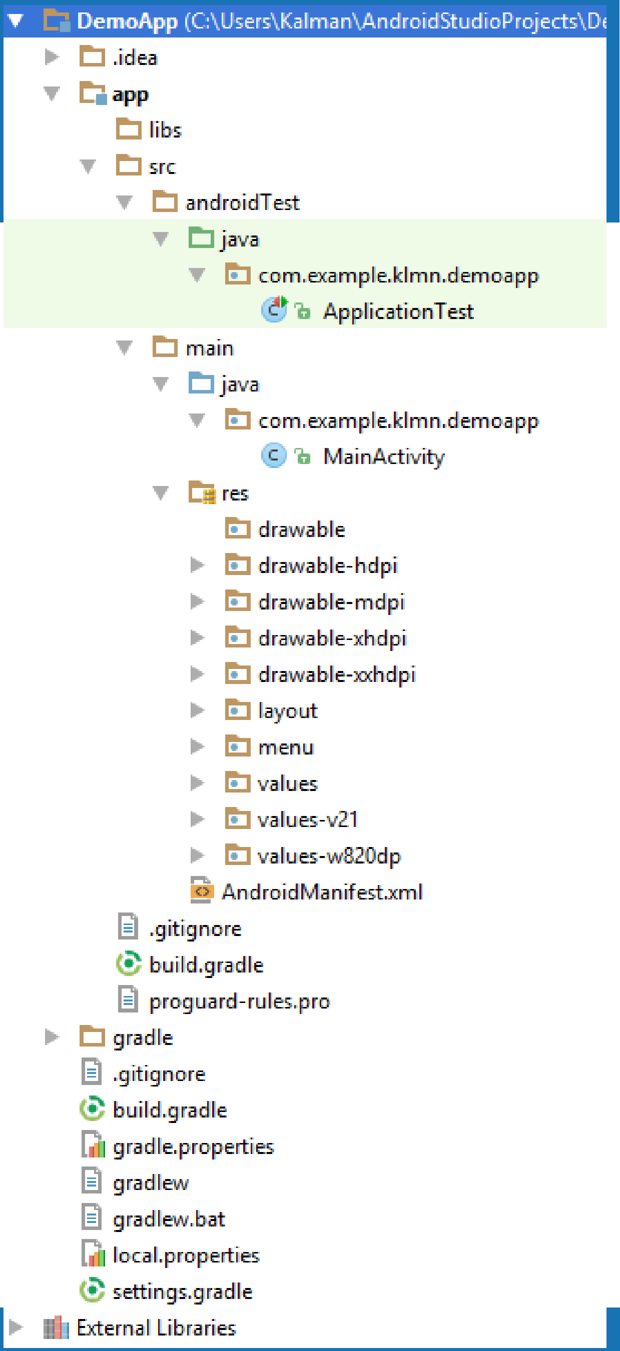


So it begins …

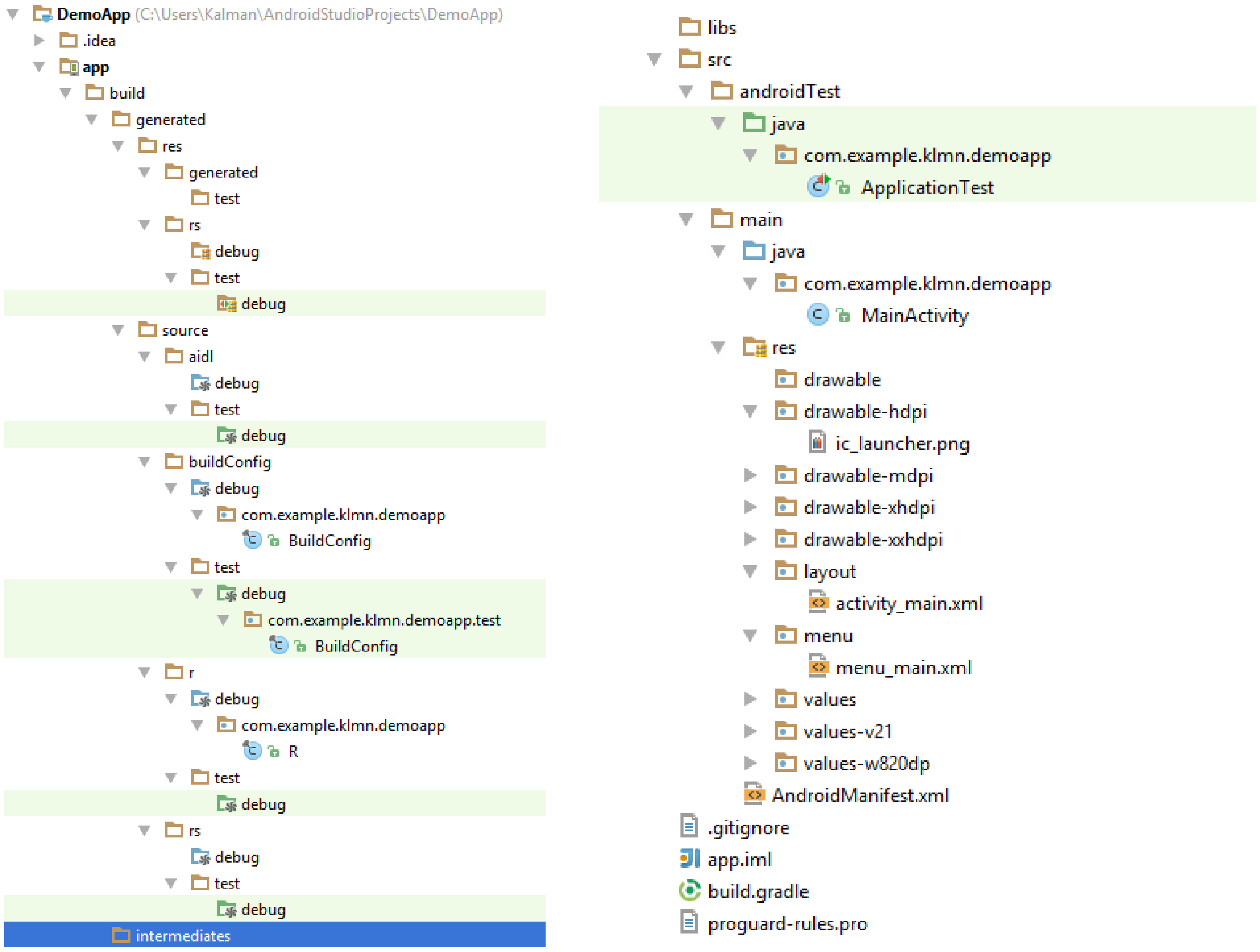
Don’t forget the details!

Project structure

- .idea
  - IntelliJ IDEA settings
- app
  - Files of Android applications
- build
  - Files generated during build
- gradle
  - Location of gradle wrapper
- build.gradle
  - Project settings for Gradle building
- gradle.properties
  - Project settings for Gradle
- gradlew or gradlew.bat
  - OS specific gradle settings
- local.properties
  - Local computer specific settings
- .iml
  - IntelliJ IDEA module information
- settings.gradle
  - Gradle tool parameters



```
DemoApp (C:\Users\Kalman\AndroidStudioProjects\De
  .idea
  app
    libs
    src
      androidTest
        java
          com.example.klmn.demoapp
            ApplicationTest
      main
        java
          com.example.klmn.demoapp
            MainActivity
        res
          drawable
          drawable-hdpi
          drawable-mdpi
          drawable-xhdpi
          drawable-xxhdpi
          layout
          menu
          values
          values-v21
          values-w820dp
        AndroidManifest.xml
    .gitignore
    build.gradle
    proguard-rules.pro
  gradle
  .gitignore
  build.gradle
  gradle.properties
  gradlew
  gradlew.bat
  local.properties
  settings.gradle
External Libraries
```



```
▼ DemoApp (C:\Users\Kalman\AndroidStudioProjects\DemoApp)
  ▶ .idea
  ▼ app
    ▼ build
      ▼ generated
        ▼ res
          ▼ generated
              test
          ▼ rs
              debug
            ▼ test
                debug
        ▼ source
          ▼ aidl
              debug
            ▼ test
                debug
          ▼ buildConfig
            ▼ debug
              ▼ com.example.klmn.demoapp
                  BuildConfig
            ▼ test
              ▼ debug
                ▼ com.example.klmn.demoapp.test
                    BuildConfig
          ▼ r
            ▼ debug
              ▼ com.example.klmn.demoapp
                  R
            ▼ test
                debug
          ▼ rs
              debug
            ▼ test
                debug
      intermediates
    libs
  ▼ src
    ▼ androidTest
      ▼ java
        ▼ com.example.klmn.demoapp
            ApplicationTest
    ▼ main
      ▼ java
        ▼ com.example.klmn.demoapp
            MainActivity
      ▼ res
          drawable
        ▼ drawable-hdpi
            ic_launcher.png
        ▶ drawable-mdpi
        ▶ drawable-xhdpi
        ▶ drawable-xxhdpi
        ▼ layout
            activity_main.xml
        ▼ menu
            menu_main.xml
        ▶ values
        ▶ values-v21
        ▶ values-w820dp
        AndroidManifest.xml
  .gitignore
  app.iml
  build.gradle
  proguard-rules.pro
```

Project structure

- build
  - Files generated after build process – flavor and version specific
  - Several builds for different API, etc.
- libs
  - User defined libraries
- src
  - androidTest
    - For Junit tests
  - main/java/ …
    - Java source codes
  - main/jni
    - Android NDK/JNI source codes
  - main/assets
    - Most of the cases it is empty
    - Files are put into the APK file, raw resources

Project structure

- src/main/res
  - anim
    - Animations encoded in XML
  - drawable (xdpi, hdpi, mdpi, ldpi)
    - Images (.jpg .png or .xml)
  - layout - *.xml
    - To describe UI layouts
  - raw
    - Resources: mp3, mp4, avi, CVS, etc.
  - values – strings.xml
    - Texts used in the application
    - Used for localization

Project structure – AndroidManifest

- src/main/AndroidManifest.xml
  - All important information about the application
  - Components
  - Hardware requirements
  - Android version compatibilities
  - Permissions
  - Java package name
  - The libraries that the application must be linked

  - http://developer.android.com/guide/topics/manifest/manifest-intro.html

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.hello"
    android:versionCode="1"
    android:versionName="1.0" >                                            <!– Application version-->

    <uses-sdk
        android:minSdkVersion="15"
        android:targetSdkVersion="19" />                                   <!– Android 4.0 and above -->

    <uses-permission android:name="android.permission.INTERNET" />         <!– can access internet -->
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />  <!– write on external storage (SD card) -->
    <uses-permission android:name="android.permission.CAMERA" />           <!– use camera -->

    <uses-feature android:name="android.hardware.camera" />                <!– requires camera -->
    <uses-feature android:name="android.hardware.camera.autofocus" />      <!– requires autofouces -->

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >                                  <!– icon, name and theme for app -->
        <activity
            android:name=".MainActivity"
            android:label="@string/app_name" >                             <!– main Activity (later) -->
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".SettingsActivity" android:screenOrientation="portrait" />
        <activity android:name=".NewsActivity" android:screenOrientation="portrait" />
    </application>
</manifest>
```

build.gradle

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 24
    buildToolsVersion "24.0.3"

    defaultConfig {
        applicationId "hu.ppke.itk.mad"
        minSdkVersion 20
        targetSdkVersion 24
        versionCode 1
        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    testImplementation 'junit:junit:4.12'
    implementation 'com.android.support:appcompat-v7:24.2.1'
}
```

Used SDK version

Package name

Minimum SDK version needed

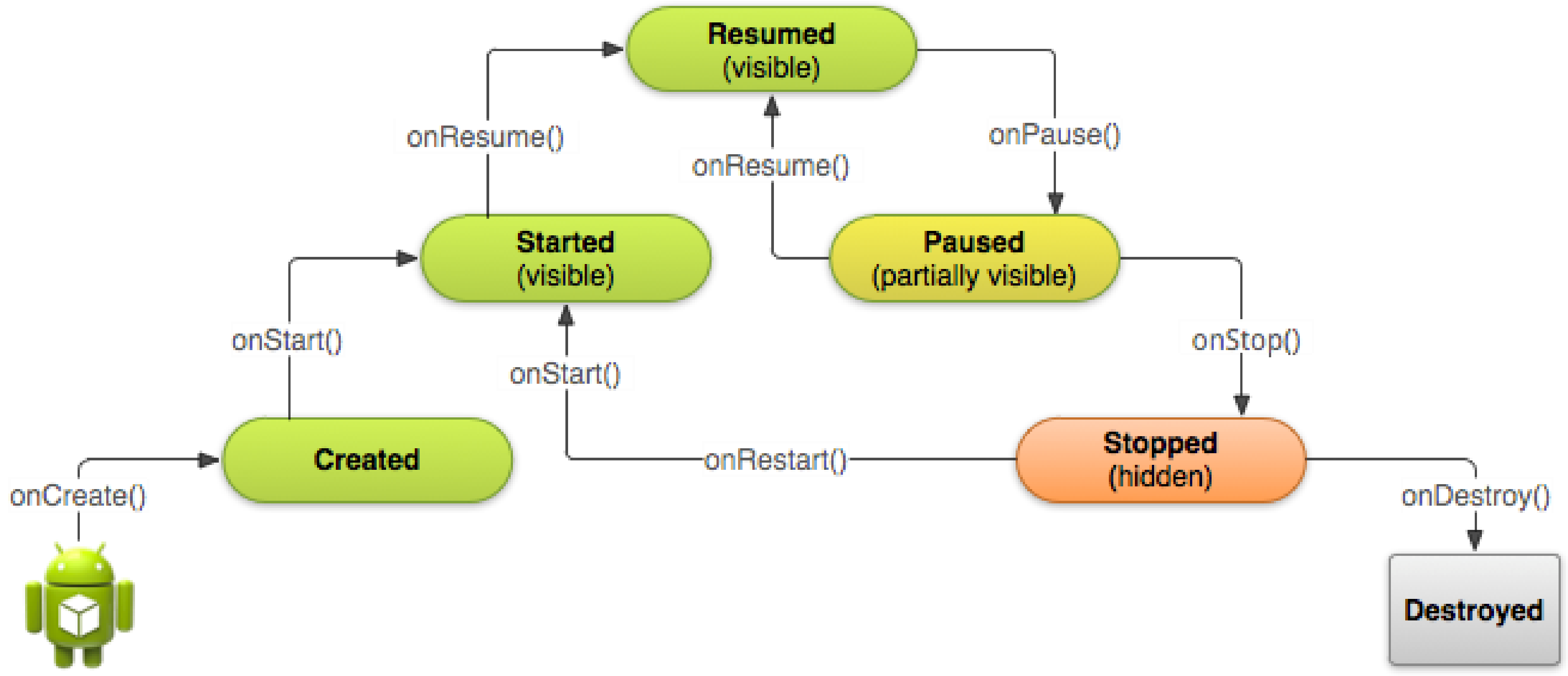
Version of the application

Used libraries

Activity

- Purposes
  - Communicate with the user
  - Handle GUI elements
  - Execute tasks
- An application can have multiple activities
- All activity is derived from `android.app.Activity` class

Activity life cycle



Activity life cycle – methods

- We are informed about the status changes of `Activity` with several different callback functions
  - We have to override these methods, and these methods are called by the system
  - Then we can execute tasks when events occur
- The life cycle functions are:
  - `onCreate:` when `Activity` starts newly (first start, or after disposal)
    - You may set the GUI and variables here
  - `onStart:` when the `Activity` is visible for the user
  - `onResume:` the `Activity` is in focus, now we can start working
  - `onPause:` when `Activity` is partially visible
    - Due to other `Activity`, or `Dialog`, …
    - In case of multi windows system (Android 7.x) when this is the inactive `Activity`
    - You may want to save the necessary information (state)
      - This have to be quick, as it blocks any other `Activity`.
      - If the `Activity` is being destroys this is the only function which execution is guaranteed!

Activity life cycle – methods

- We are informed about the status changes of `Activity` with several different callback functions
  - We have to override these methods, and these methods are called by the system
  - Then we can execute tasks when events occur
- The life cycle functions are:
  - `onCreate:` when `Activity` starts newly (first start, or after disposal)
    - You may set the GUI and variables here
  - `onStart:` when the `Activity` is visible for the user
  - `onResume:` the `Activity` is in focus, now we can start working
  - `onPause:` when `Activity` is partially visible
    - Due to other `Activity`, or `Dialog`, …
    - In case of multi windows system (Android 7.x) when this is the inactive `Activity`
    - You may want to save the necessary information (state)
      - This have to be quick, as it blocks any other `Activity`.
      - If the `Activity` is being destroys this is the only function which execution is guaranteed!

Activity life cycle – methods

- `onStop:` when the `Activity` is invisible
  - It is totally invisible due to another `Activity`, or any other reason
    - Incoming call
    - Screen lock
- `onDestroy:` when `finish()` is called, or memory is needed
  - The `Activity` is destroyed (killed, deleted, …)
  - If the memory is needed instantly then this call may be discarded.
  - Do not save data here, only set the affected variables to `null`

- In all life cycle callback method you have to call the superclass' same method
  - Example: `super.onCreate`
  - The Android system check it
  - Runtime Exception is thrown if you violate this rule

Screen layouts

- You can define the screens two ways
  - Static method
    - Creating .xml files in the res/layout folder
  - Dynamic method
    - In the java source code
    - Creating new instances of View elements
- The layout defines the positions, sizes of elements in the screen
- A layout class is derived from the `View` class!

Attributes of GUI elements

- `layout_width` and `layout_height`
  - Specify the width and height of the view element or layout
    - It is required to specify
    - Runtime exception is thrown if it is missing
  - The actual size is calculated (based on this value and other elements)
- Possible values
  - `wrap_content` – as the content requires
  - `match_parent` – the size of this element is specified by the parent
  - fix size – the unit is dp, which is the devices independent pixel
- `id`: optional (you have to specify if you wish to access it from `Activity`)
- `gravity`: the view is aligned
  - `left`, `right`, `bottom`
  - `center` – vertical and horizontal
    - `horizontal`, `vertical`
    - You can mix: `android:gravity="center|bottom"`

Attributes of GUI elements

- `layout_weight="2"`
  - The „importance" of the element can be set
  - More important element can „push" aside the other elements
    - There are three views but the middle should be larger
- `visibility`:
  - visible – you can see it, visible
  - invisible – cannot be seen, but its size is considered
  - gone – cannot be seen, and no space is occupied
- `padding`
  - Space between the elements
- `background`
  - Could be a color or drawing
- There are attributes which are depending on the actual class of the parent `ViewGroup`
  - For example: the column of a table can be interpreted only in a table

GUI elements

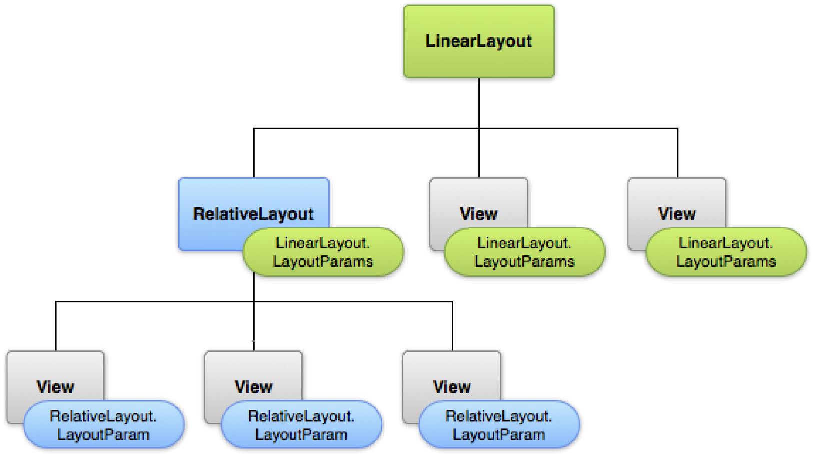
- Layouts
  - Linear Layout
  - Relative Layout
  - Constraint Layout
  - Coordinator Layout
  - Recycler View
  - Frame Layout
  - Web View
- Widgets and other Views
  - Text View
  - Edit Text
  - Auto Complete Text View
  - Button
  - Image View
  - Scroll View
  - View Pager
  - Map View
  - etc

GUI structure

- The GUI is built from Widgets which are `View` and `ViewGroup` elements arranged in a tree structure
  - The `ViewGroup` is extended from the `View` class also
  - The `ViewGroup` is a special `View`, which can have children, so it can contain other elements
- It is possible the define own Views or View groups, but there are a lot of predefined one.
  - If you need to create an own view Extend from the proper class

View hierarchy

- There is one root element
- Set the root element with the `setContentView` function of the Activity class.
  - In the onCreate() function
- `Every ViewGroup` responsible for the drawing of it's children
- Views are drawn on the top of root.
- We can add child to a ViewGroup dynamically with the `addView(View)` `function`



Inflation

- The hierarchy can be derived in xml files as well
  - In that case the parameter of the `setContentView` is not a View, but an int
    - This is an id for the layout file
    - The id and the xml are connected in the `R.java` file
    - The connection is automatically created
  - First, the system creates the view hierarchy based on the layout
  - Then the it calls the `setContentView(View)` function
  - Example:
    - Hello world application
    - `setContentView(R.layout.activity_main);`

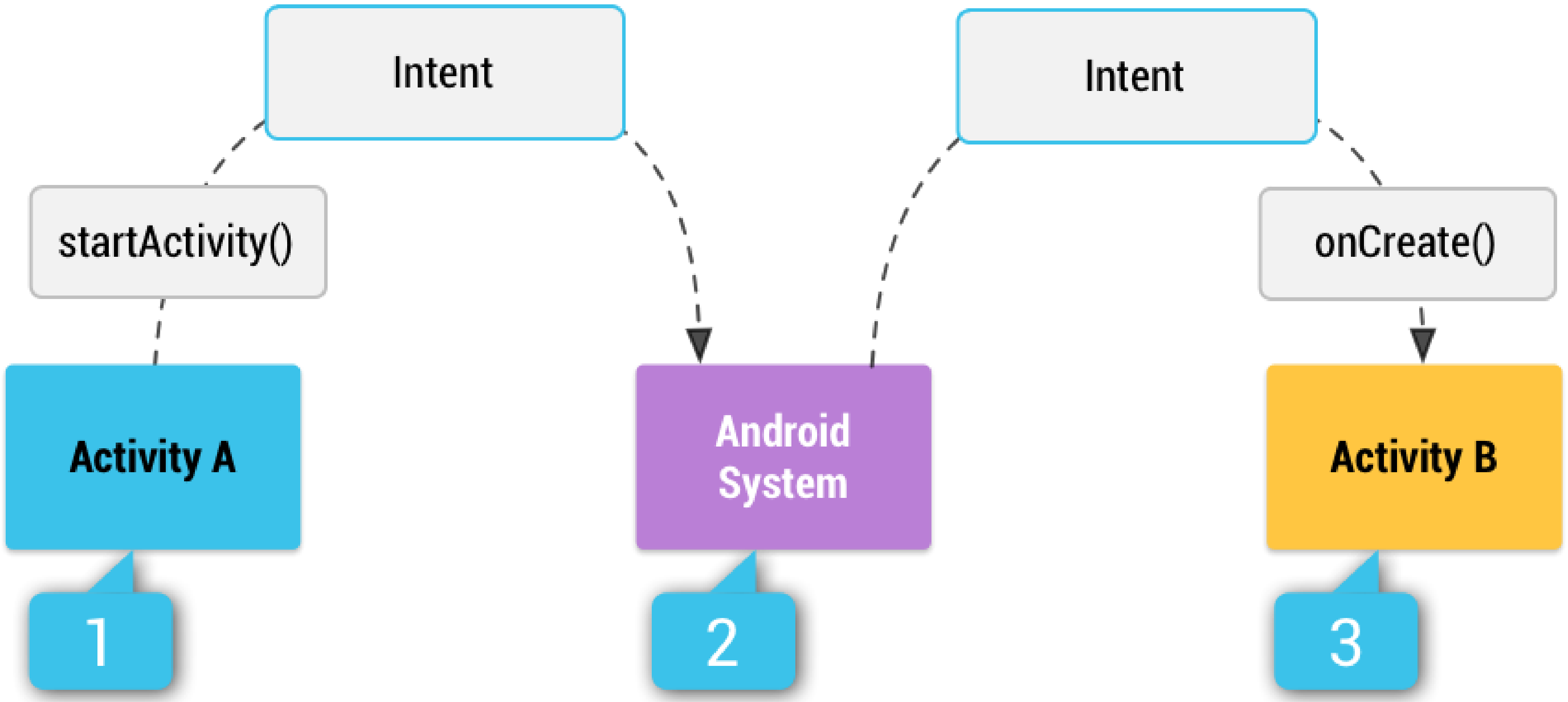
The new stuff is coming

Right now!

Intra-application communication

- We already had one `Activity`
  - Which was designed to represent a set of „real-life" actions
  - It have been used as entry point(s) for the application
    - This has not been detailed previously
- However, an application is a set of `Activity`
  - They are working together to solve tasks
  - The communication between `Activity`-s have to be implemented
    - `Intent`-s are used for this purpose

Intents


Intent
startActivity()
Activity A
1
Android System
2
Intent
onCreate()
Activity B
3

Intent

- An `Intent` is used for connecting the components of application
  - We would like to start another Activity – to perform another tasks
  - An `Intent` object is used to describe our intentions abstractly
    - To describe the task or set of tasks which are wanted to be performed
  - Application components can be bonded in runtime
  - „Applications without borders"
    - Intents can be used between applications
- An `Intent` can be either
  - Explicit
    - The `Activity` / `Service` to be started is specified explicitly
    - For example, it is used to start an `Activity` in our applications
  - Implicit
    - Our intentions are specified more abstractly (the application component is not specified explicitly)
    - For example „Send"
      - There may be several application components which can be started:
        - Email
        - SMS
        - Bluetooth
        - …

Content of an `Intent`

- Name of component, which is wanted to be started
  - Optional
  - If it is set the `Intent` is explicit, otherwise implicit
  - In case of `Service`, it is mandatory
    - So a service can be started only with explicit intent
  - `setComponent()`, `setClass()`, `setClassName()`, or constructor
- `Action` – A `String`, which specifies the task
  - There are predefined ones, but we also can define new ones
    - `static final String ACTION_TIMETRAVEL = "com.example.action.TIMETRAVEL";`
- `Data` – A URI, to specify data or MIME type
  - The data type depends on the activity to be started
  - Most of the cases it is mandatory, as it contains important information for the receiving component
  - `setData()`, `setType()`, `setDataAndType()`

Content of an `Intent`

- `Category` – `String` to provide more information about the `Intent,` and to specify what components can be involved
  - Arbitrary number of categories can be set
  - Optional
  - `addCategory()`
- `Extras` – Key-value pairs to provide specific information for receiving components
  - For example: in case of email: the text and recipients of the email
  - `putExtra()`, `putExtras()`
  - The are predefined keys, but we can define new ones.
  - `static final String EXTRA_GWS = "com.example.EXTRA_GIGAWATTS";`
    - Note that, that this string starts with the package name of the application in order to avoid overlapping between different applications
- `Flags` – Further metadata to specify how the `Intent` should be processed
  - `setFlag()`

Example

- Explicit `Intent`:

  -
    ```
    Intent downloadIntent = new Intent(this, DownloadService.class);
    downloadIntent.setData(Uri.parse(fileUrl));
    startService(downloadIntent);
    ```
  - The first parameter of the constructor is the `Context`, the second is the target component.
- Implicit `Intent`

  -
    ```
    Intent sendIntent = new Intent();
    sendIntent.setAction(Intent.ACTION_SEND);
    sendIntent.putExtra(Intent.EXTRA_TEXT, textMessage);
    sendIntent.setType("text/html");
    ```
- Use the following way to send an implicit `Intent` securely.
  - The intent is only sent when at least one matching component exists.

  -
    ```
    if (sendIntent.resolveActivity(getPackageManager()) != null) {
      startActivity(sendIntent);
    }
    ```

Sending `Intents`

- `Activity`
  - `Context.startActivity(Intent)`
  - `Activity.startActivityForResult(Intent, int)`
- `Service`
  - `Context.startService(Intent)`
  - `Context.bindService(Intent, ServiceConnection, int)`
- `BroadcastReciever`
  - `Context.sendBroadcast(Intent)`
- Different calls do not overlap each other
  - An `Intent` sent with `startService` delivered to services only.
- In all three cases, the Android systems determines the most appropriate component which is capable of receiving the `Intent`

Exemplary `Action/Data` pairs

- `ACTION_VIEW` content://contacts/people/1
  - Retrieve the person with id=1 from contacts
- `ACTION_DIAL` content://contacts/people/1
- `ACTION_VIEW` tel:123
  - Show a dialer with the given phone number
- `ACTION_EDIT` content://contacts/people/1
  - Edit the data of contact with id=1
- `ACTION_VIEW` content://contacts/people/
  - Entire contact list

Action

Standard

- ACTION_MAIN
- ACTION_VIEW
- ACTION_ATTACH_DATA
- ACTION_EDIT
- ACTION_PICK
- ACTION_CHOOSER
- ACTION_GET_CONTENT
- ACTION_DIAL
- ACTION_CALL
- ACTION_SEND
- ACTION_SENDTO
- ACTION_ANSWER
- ACTION_INSERT
- ACTION_DELETE
- ACTION_RUN
- ACTION_SYNC
- ACTION_PICK_ACTIVITY
- ACTION_SEARCH
- ACTION_WEB_SEARCH
- ACTION_FACTORY_TEST

Broadcast

- ACTION_TIME_TICK
- ACTION_TIME_CHANGED
- ACTION_TIMEZONE_CHANGED
- ACTION_BOOT_COMPLETED
- ACTION_PACKAGE_ADDED
- ACTION_PACKAGE_CHANGED
- ACTION_PACKAGE_REMOVED
- ACTION_PACKAGE_RESTARTED
- ACTION_PACKAGE_DATA_CLEARED
- ACTION_UID_REMOVED
- ACTION_BATTERY_CHANGED
- ACTION_POWER_CONNECTED
- ACTION_POWER_DISCONNECTED
- ACTION_SHUTDOWN

Categories

- CATEGORY_DEFAULT
- CATEGORY_BROWSABLE
- CATEGORY_TAB
- CATEGORY_ALTERNATIVE
- CATEGORY_SELECTED_ALTERNATIVE
- CATEGORY_LAUNCHER
- CATEGORY_INFO
- CATEGORY_HOME
- CATEGORY_PREFERENCE
- CATEGORY_TEST
- CATEGORY_CAR_DOCK
- CATEGORY_DESK_DOCK
- CATEGORY_LE_DESK_DOCK
- CATEGORY_HE_DESK_DOCK
- CATEGORY_CAR_MODE
- CATEGORY_APP_MARKET

Extras

- EXTRA_ALARM_COUNT
- EXTRA_BCC
- EXTRA_CC
- EXTRA_CHANGED_COMPONENT_NAME
- EXTRA_DATA_REMOVED
- EXTRA_DOCK_STATE
- EXTRA_DOCK_STATE_HE_DESK
- EXTRA_DOCK_STATE_LE_DESK
- EXTRA_DOCK_STATE_CAR
- EXTRA_DOCK_STATE_DESK
- EXTRA_DOCK_STATE_UNDOCKED
- EXTRA_DONT_KILL_APP
- EXTRA_EMAIL
- EXTRA_INITIAL_INTENTSEXTRA_INTENT
- EXTRA_KEY_EVENT
- EXTRA_ORIGINATING_URI
- EXTRA_PHONE_NUMBER
- EXTRA_REFERRER
- EXTRA_REMOTE_INTENT_TOKEN
- EXTRA_REPLACING
- EXTRA_SHORTCUT_ICON
- EXTRA_SHORTCUT_ICON_RESOURCE
- EXTRA_SHORTCUT_INTENT
- EXTRA_STREAM
- EXTRA_SHORTCUT_NAME
- EXTRA_SUBJECT
- EXTRA_TEMPLATE
- EXTRA_TEXT
- EXTRA_TITLE
- EXTRA_UID

Extra

- Putting extra into an intent object

  -
    ```
    Intent i = new Intent(context, SendMessage.class);
    i.putExtra("id", user.getUserAccountId());
    i.putExtra("name", user.getUserFullName());
    context.startActivity(i);
    ```
- Retrieve extra information from intent object

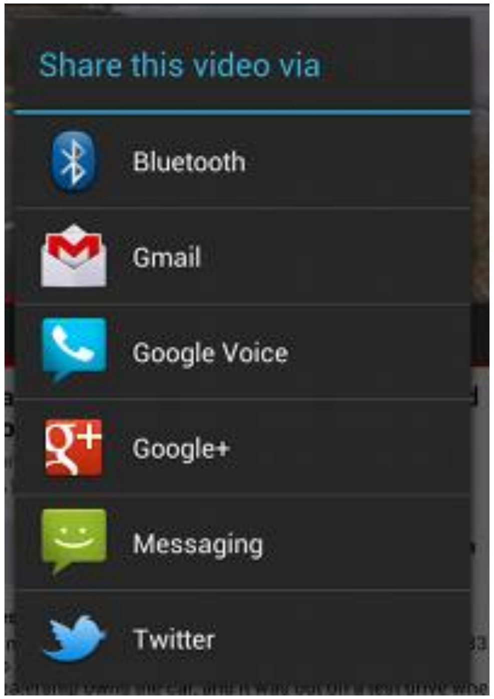
  -
    ```
    Intent intent = getIntent(); // If it is not received as parameter
    String id = intent.getStringExtra("id");
    String name = intent.getStringExtra("name");
    ```
- Alternative method

  -
    ```
    Bundle extras = getIntent().getExtras();
    String userName;
    if (extras != null) {
      userName = extras.getString("name");
    }
    ```

Implicit case

- When an `Intent` can be received by multiple `Activity-`s, then user can make the choice on a pop-up view:
  - The appearance of this view can be enforced:

```
Intent sendIntent = new Intent(Intent.ACTION_SEND);
// ...
String title =
getResources().getString(R.string.chooser_title);
Intent chooser = Intent.createChooser(sendIntent,
title);
if (sendIntent.resolveActivity(getPackageManager())
!= null) {
  startActivity(chooser);
}
```



`IntentFilter`

- For a component, it can be specified what `Intent`-s can be accepted by that component
  - `IntentFilters` are used for this task
  - Multiple filters can be specified for one component
    - Logical OR is used between them
  - Filters are specified in the *AndroidManifest.xml* file, most of the cases
    - As Android system must know the filters before the application is started
    - The starting activity (the entry point of the application) is also specified with an intent filter
    - However, it can be also specified from code, using the `IntentFilter` class
- Only used for implicit `Intent` invocation
  - As in case of explicit `Intent` there is no need to determine the corresponding component

Fields of filters

- According to the fields of intents the following fields of filters exist
    - Action
    - Data
    - Category

```
<intent-filter>
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.LAUNCHER" />
</intent-filter>

<intent-filter>
    <action android:name="android.intent.action.VIEW" />
    <action android:name="android.intent.action.EDIT" />
    <action android:name="android.intent.action.PICK" />

    <category android:name="android.intent.category.DEFAULT" />

    <data android:mimeType="vnd.android.cursor.dir/vnd.google.note" />
</intent-filter>
```

Evaluation of `IntentFilter`

- Three tests are performed
  - Action
    - There may be more `Action` in the filter
      - `Intents` containing any of specified `Action` in `IntentFilter` will match
    - If a filter does not specify any `Action`, then none of the `Intents` matches
    - Any `IntentFilter` without Action matches to `IntentFilters` containing at least one `Action`

```
<intent-filter>
    <action android:name="android.intent.action.EDIT" />
    <action android:name="android.intent.action.VIEW" />
</intent-filter>
```

    - That is when we define what we would like to do, then that components can be started which are defined as they capable of performing the specific task
    - Otherwise, if the task is not specified, then any component can be started which can perform any task

Evaluation of `IntentFilter`

- Three tests are performed
  - Category
    - All categories defined in the `Intent` have to be enumerated in the intent filter
      - The intent filter may contain more
    - If `Intent` does not specify category, then it will match to the filter
      - To receive implicit intent, the `android.intent.category.DEFAULT` have to be specified in the filter as the `startActivity` call puts this category to the intent
    - In the case of activities which can be started from the app launcher, the following category has to be specified: `"android.intent.category.LAUNCHER"`

```
<intent-filter>
    <category android:name="android.intent.category.DEFAULT" />
    <category android:name="android.intent.category.BROWSABLE" />
</intent-filter>
```

- As a result, all the categories have to be supported by the component, that are requested in the intent
- Furthermore, in the case of an `Activity,` the `DEFAULT` must be specified as well

Evaluation of `IntentFilter`

- Three tests are performed
  - Data
    - The data specified in the `Intent` should be matched to any of the data specified in the filter
    - Each `<data>` tag specifies a `Uri`
    - Attributes: scheme, host, port, and path
    - In `Uri` : scheme://host:port/path
    - If there are no data defined in the `Intent`, then it will match to filters without data
    - If the `Intent` has `Uri` but no data type, then it will match to filters without type and the `Uri` matches
    - If the `Intent` defines type without `Uri`, then it will match to filters with the same type and without `Uri`

```
<intent-filter>
    <data android:mimeType="video/mpeg" android:scheme="http" />
    <data android:mimeType="video/mp4" android:scheme="http" />
</intent-filter>
```

- As a result, the parameters of data sent with the Intent must match perfectly to the intent filter (to receive only compatible data)

Evaluation of `IntentFilter`

- scheme://host:port/path
  - If no scheme specified, then the host is ignored
  - If no host specified, then the port is ignored
  - If neither scheme nor host specified, then the path is ignored
- When the URI is compared to the filter only that part is used which is specified in the filter
  - If filter specifies only scheme then URI with the same scheme matches
  - If both scheme and host:port are specified, then the path is ignored, but scheme and host:port have to match
  - Otherwise, all components have to match

Launcher example

```
<activity android:name=".MainActivity" >
  <intent-filter>
    <action android:name="android.intent.action.MAIN" />

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

Example

```
 <activity android:name=".ShareActivity">
    <!-- Activity handles "SEND" actions with text data -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="text/plain"/>
    </intent-filter>
    <!-- Activity handles "SEND" and "SEND_MULTIPLE" with media data -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <action android:name="android.intent.action.SEND_MULTIPLE"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="application/vnd.google.panorama360+jpg"/>
        <data android:mimeType="image/*"/>
        <data android:mimeType="video/*"/>
    </intent-filter>
</activity>
```

startActivity()

- `startActivity()` is used to launch a new `Activity`
- Example (as we seen previously)

  -
    ```
    Intent sendIntent = new Intent();
    sendIntent.setAction(Intent.ACTION_SEND);
    sendIntent.putExtra(Intent.EXTRA_TEXT, textMessage);
    sendIntent.setType("text/html");
    startActivity(sendIntent);
    ```

- This is not the sole possibility!

startActivityForResult()

- An `Activity` can be started by the `startActivityForResult()` which returns the result of the started activity
  - The type of the function is **`void`** – as this is not a synchronous call
    - Note that it is also possible that the originating `Activity` may be disposed (to free up resources)
- Example

```
static final int REQUEST = 1;

private void pickContact() {
  Intent intent = new Intent(Intent.ACTION_PICK, Uri.parse("…"));
  intent.setType(Phone.CONTENT_TYPE);
  startActivityForResult(intent, REQUEST);
}
```

- The invoked `Activity` ends with calling its `finish()` function.
  - The result is put into an `Intent`
  - The result is set by calling **`this`**`.setResult()`

startActivityForResult()

- Receiving the result
  - As the invoked activity ends and the originating activity is in the foreground then the `onActivityResult()` function is called
  - Parameters
    - Original request code
    - Result code
      - `RESULT_OK` or `RESULT_CANCELLED`, depending on how the invoked `Activity` finished
    - The result `Intent`
- Example
-

```
@Override
protected void onActivityResult(int reqCode, int resCode, Intent data) {
  if (reqCode == REQUEST) {
    if (resCode == RESULT_OK) {
      /* ... */
    }
  }
}
```

Rotation of the screen

- The screen can be rotated at any time!
- In this case, the layout needs to be replaced with its contents
- We can define different layouts for the portrait and landscape rotation so it will be replaced automatically (the how-to is on a later class)
- But the Activity object will be restarted with the view! Its lifecycle will restart again
- We need to store and restore the state of the Activity
  - For storing the state we can use the `onSaveInstanceState` `method`, for restoring the `onCreate` or the `onRestoreInstanceState` `methods.`
  - It is possible to turn off this automatic behavior, but then the orientation change needs to be handled by ourselves (`android:configChanges="orientation"` *in the manifest*)
  - Or we can lock the orientation of the Activity, so it won't react for the orientation changes (`android:screenOrientation="portrait"` in the manifest)

Homework

- Create an Android application which has two Activities:
  - In the first Activity display a list of a groceries list and a TextView.
    - The items in the list should contain a name at least, and they should be clickable
    - The TextView should display the number of the items in the shopping cart
      - The items can be placed into the cart in the second activity
    - The items that are in the cart should have a different color in the list
    - When a list item is clicked start the second Activity
  - The second Activity displays the name of the clicked (selected) item and a Button.
    - When the button is pressed, the first Activity should be opened again, and the selected item should be placed into the cart
    - If the back button is pressed, go back to the first Activity (and don't place the animal to the basket)
    - An item can be placed to the cart only once (this is a limitation on the number of items)
    - When an item is selected, which is already in the cart, the button should have a different text, and it removes the item from cart
  - (use the startActivityForResult method and store all data in the first Activity)

Service, BroadcastReceiver, Multithreading, etc.

Next week