ITMC311 Introduction to Mobile Application Development

Introduction to

- Android Studio
- Mobile Application Components

Android Studio

- Google's official Android IDE, in v1.0 as of November 2014
  - replaces previous Eclipse-based environment
  - based on IntelliJ IDEA editor
  - free to download and use
  - Java SE Development Kit 8

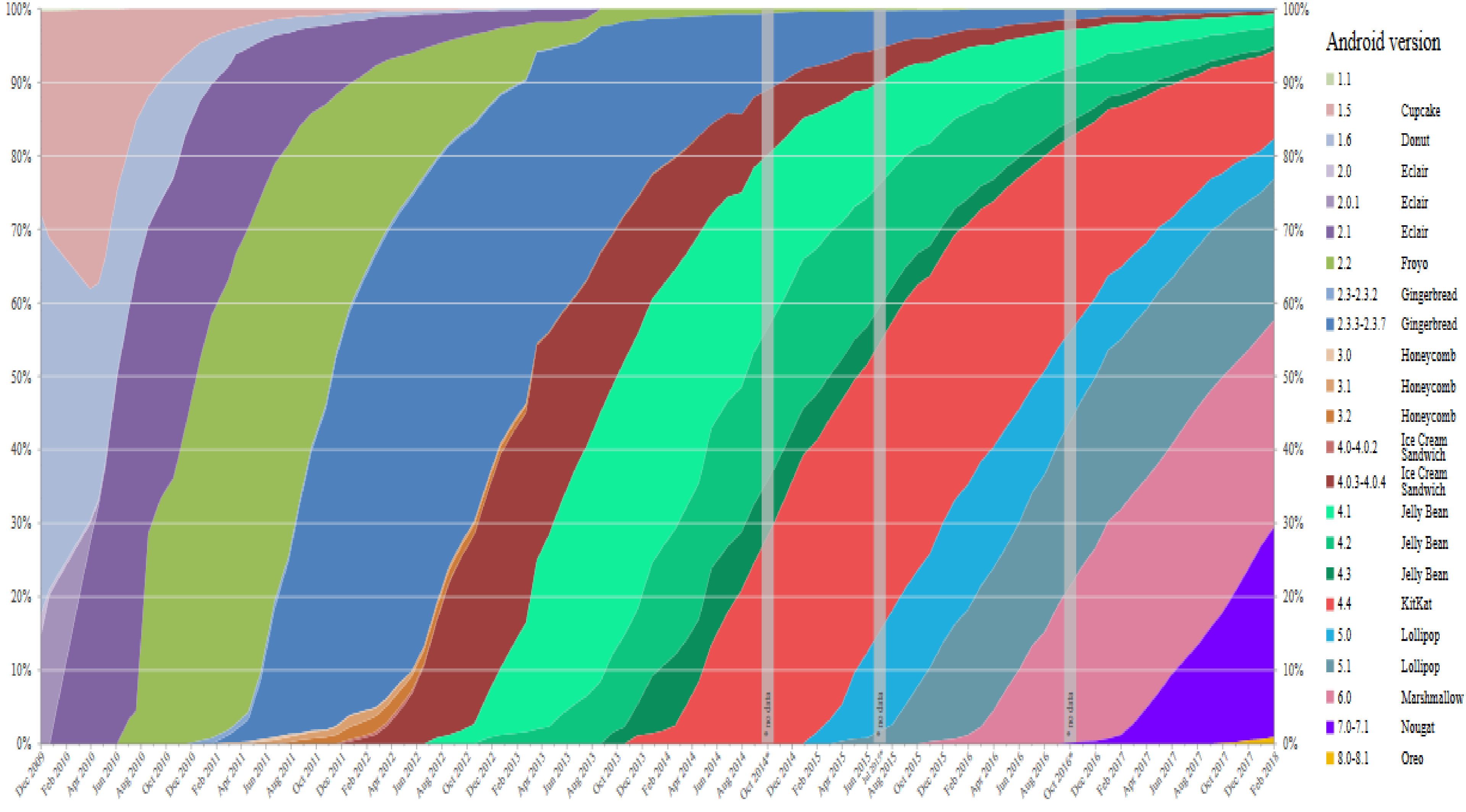
- What is Android?
  - originally purchased from Android, Inc. in 2005
  - runs on phones, tablets, watches, TVs, ...
  - based on **Java** (dev language) and **Linux** (kerne
  - and now #1 overall OS worldwide!

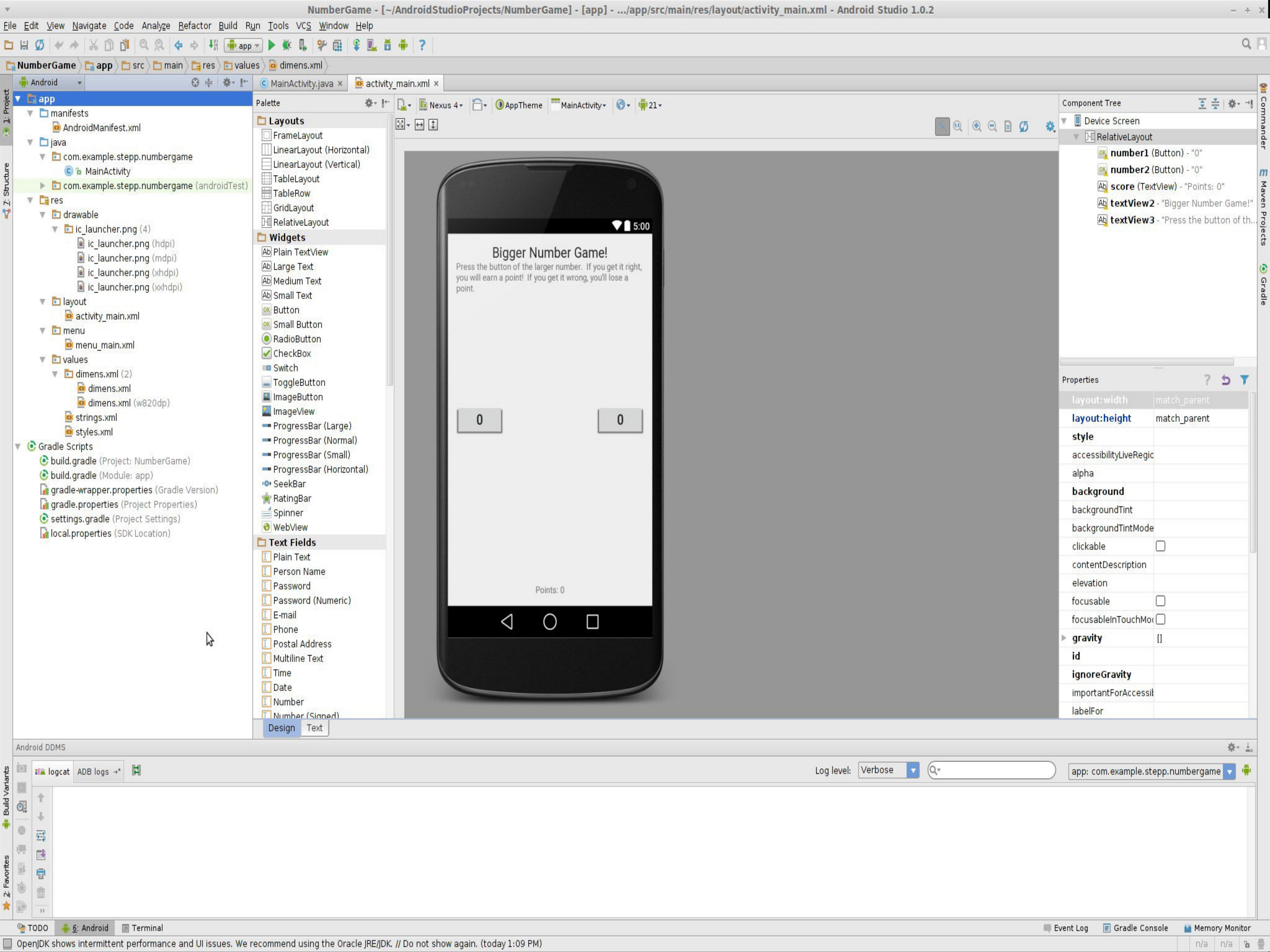
HTC Dream Oct 08

Android version history

Version	API level	Date	Name
1.0-1.1	1,2	Sep 2008	none
1.5	3	Apr 2009	Cupcake
1.6	4	Sep 2009	Donut
2.0-2.1	5,6,7	Oct 2009	Eclair
2.2	8	May 2010	Froyo
2.3	9,10	Dec 2010	Gingerbread
3.0	11,12,13	Feb 2011	Honeycomb
4.0	14,15	Oct 2011	Ice Cream Sandwich
4.1-4.3	16,17,18	Jun 2012	Jelly Bean
4.4	19,20	Sep 2013	Kit Kat
5.0-5.1	21, 22	Jun 2014	Lollipop
6.0	23	May 2015	Marshmallow
7.0-7.1	24-25	August 2016	Nougat
8.0	26-27	August 2017	Oreo
9.0	28	August 2018	Pie

Global Android version distribution since December 2009, as of October 2018





NumberGame - [~/AndroidStudioProjects/NumberGame] - [app] - .../app/src/main/res/layout/activity_main.xml - Android Studio 1.0.2

5:00

Bigger Number Game!

Press the button of the larger number. If you get it right, you will earn a point! If you get it wrong, you'll lose a point.

0 0

Points: 0

Component Tree

- Device Screen
  - RelativeLayout
    - number1 (Button) - "0"
    - number2 (Button) - "0"
    - score (TextView) - "Points: 0"
    - textView2 - "Bigger Number Game!"
    - textView3 - "Press the button of th...

Properties

layout:width	match_parent
layout:height	match_parent
style	
accessibilityLiveRegic	
alpha	
background	
backgroundTint	
backgroundTintMode	
clickable	□
contentDescription	
elevation	
focusable	□
focusableInTouchMo	□
gravity	[]
id	
ignoreGravity	
importantForAccessil	
labelFor	

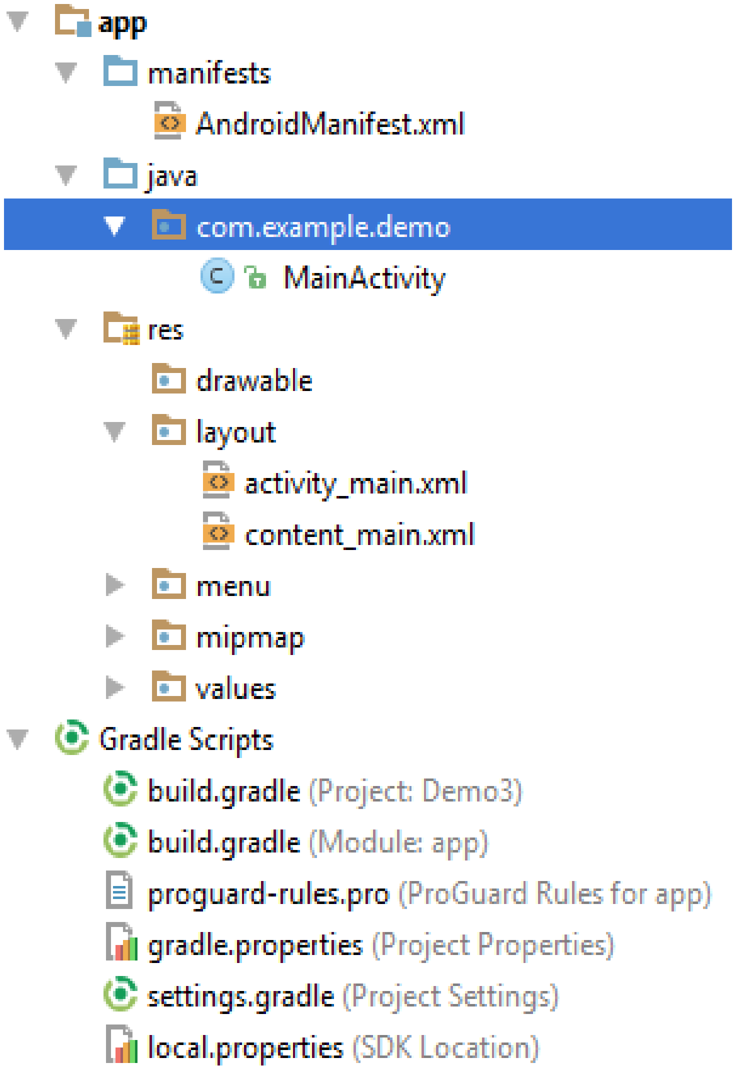
Android DDMS

Log level: Verbose

app: com.example.stepp.numbergame

OpenJDK shows intermittent performance and UI issues. We recommend using the Oracle JRE/JDK. // Do not show again. (today 1:09 PM)

Project structure



```
▼ app
    ▼ manifests
          AndroidManifest.xml
    ▼ java
        ▼ com.example.demo
              MainActivity
    ▼ res
          drawable
        ▼ layout
              activity_main.xml
              content_main.xml
        ▶ menu
        ▶ mipmap
        ▶ values
▼ Gradle Scripts
      build.gradle (Project: Demo3)
      build.gradle (Module: app)
      proguard-rules.pro (ProGuard Rules for app)
      gradle.properties (Project Properties)
      settings.gradle (Project Settings)
      local.properties (SDK Location)
```

App manifest

Java code

Resources

Build scripts

Android Manifest – example.

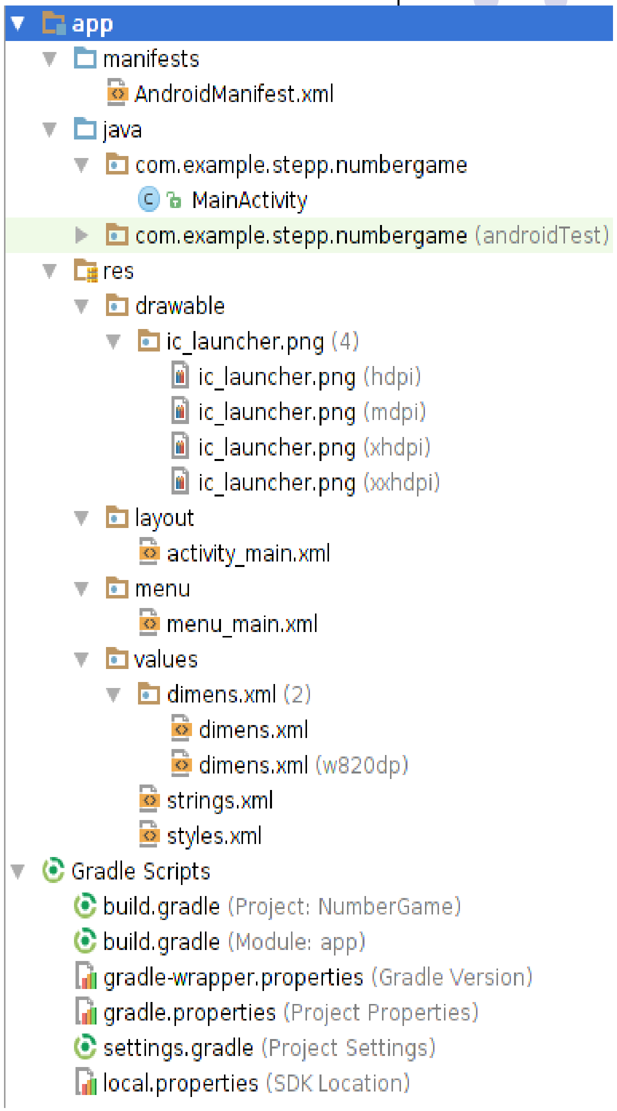
- Its main purpose in life is to declare the components to the system:
  - Overall project config and settings.
  - Every application must have an **AndroidManifest.xml** file.
  - Contains essential **information** the Android system must have before it can run any of the app's code.

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.demo">
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name="com.example.demo.MainActivity"
            android:label="@string/app_name"
            android:theme="@style/AppTheme.NoActionBar">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

Project structure (Cont.)

- **src/java/...**
  - source code for your Java classes
  - **Example: MainActivity.java**
    - determines the Activity's behavior
- **res/...** = resource files *(many are **XML**)*
  - drawable/ = images
  - layout/ = descriptions of GUI layout
  - menu/ = overall app menu options
  - values/ = constant values and arrays
  - strings = localization data
  - styles = general appearance styling
  - **Example: Main_Activity.xml**
    - determines most of the visual appearance
    - resource files *(many are **XML**)*

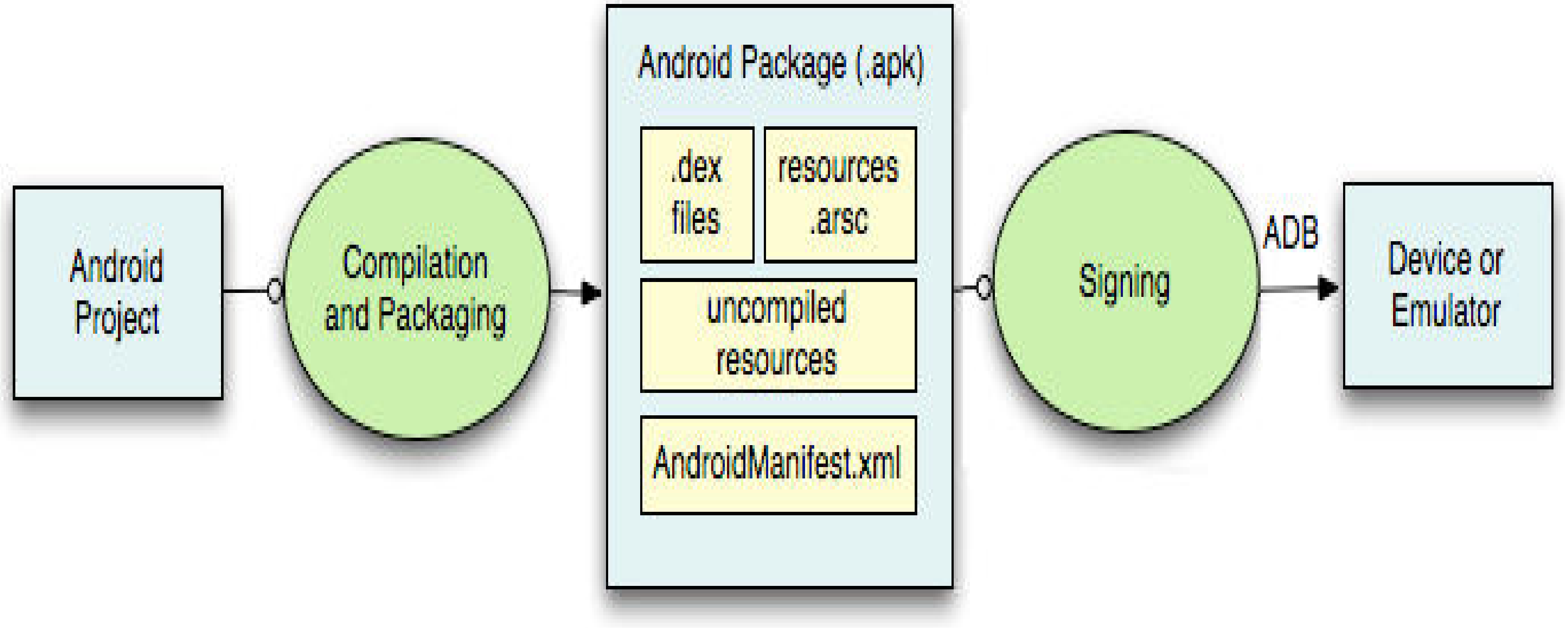


```
▼ app
  ▼ manifests
      AndroidManifest.xml
  ▼ java
    ▼ com.example.stepp.numbergame
        MainActivity
    ▶ com.example.stepp.numbergame (androidTest)
  ▼ res
    ▼ drawable
      ▼ ic_launcher.png (4)
          ic_launcher.png (hdpi)
          ic_launcher.png (mdpi)
          ic_launcher.png (xhdpi)
          ic_launcher.png (xxhdpi)
    ▼ layout
        activity_main.xml
    ▼ menu
        menu_main.xml
    ▼ values
      ▼ dimens.xml (2)
          dimens.xml
          dimens.xml (w820dp)
        strings.xml
        styles.xml
▼ Gradle Scripts
    build.gradle (Project: NumberGame)
    build.gradle (Module: app)
    gradle-wrapper.properties (Gradle Version)
    gradle.properties (Project Properties)
    settings.gradle (Project Settings)
    local.properties (SDK Location)
```

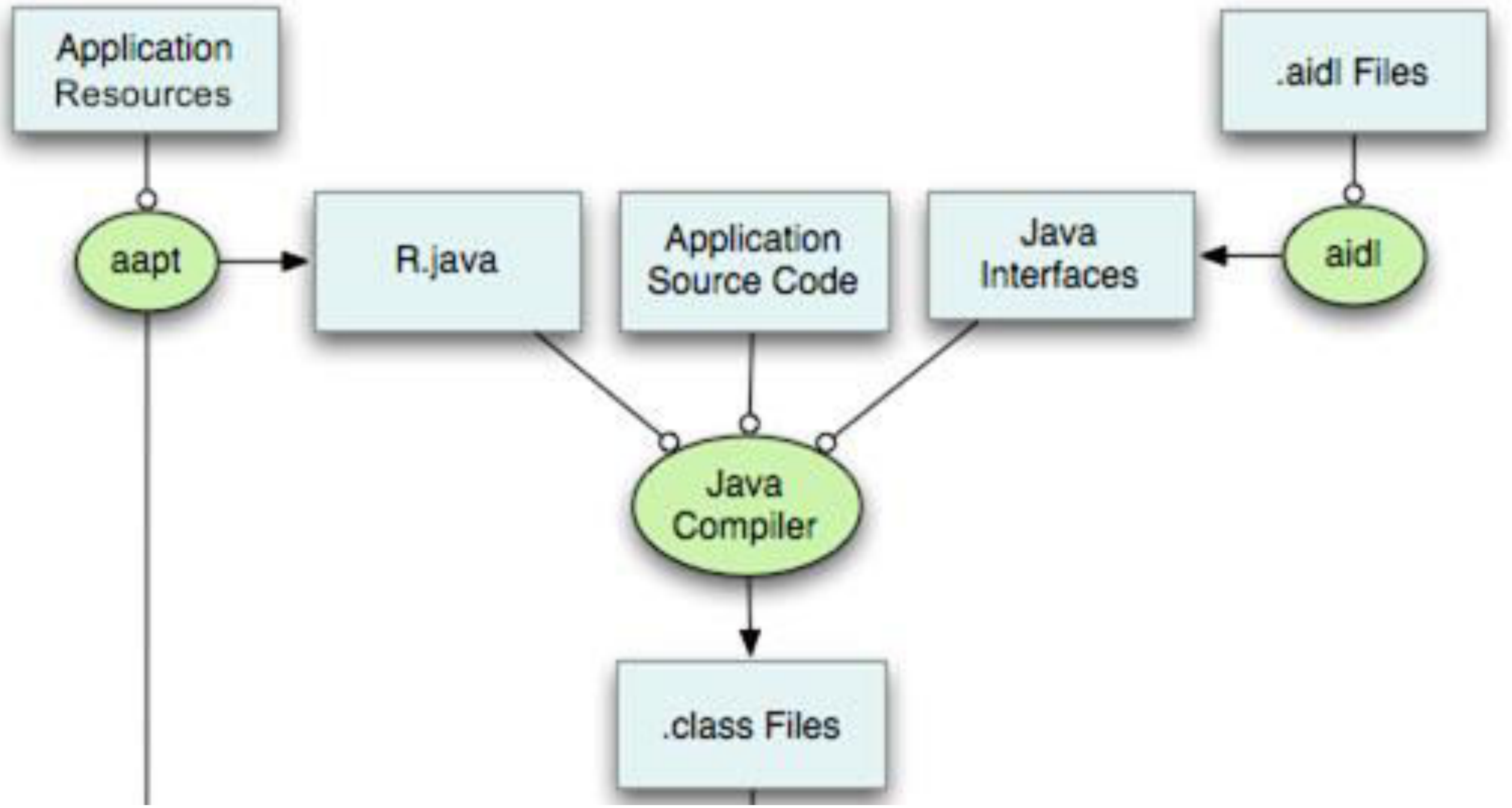
What is Android build system (Gradle)?

- **Gradle**
  - a build/compile management system
  - **Example: build.gradle** = main build config file
- It is the toolkit you use to build, test, run and package your apps.
- It can run as an integrated tool from the Android Studio menu and independently from the command line.
- You can use the features of the build system to:
  - Customize, configure, and extend the build process.
  - Create multiple APKs for your app with different features using the same project and modules.
  - Reuse code and resources across source sets.

The components involved in building and running an app



Step-1: Android build system



Step-1: Continued

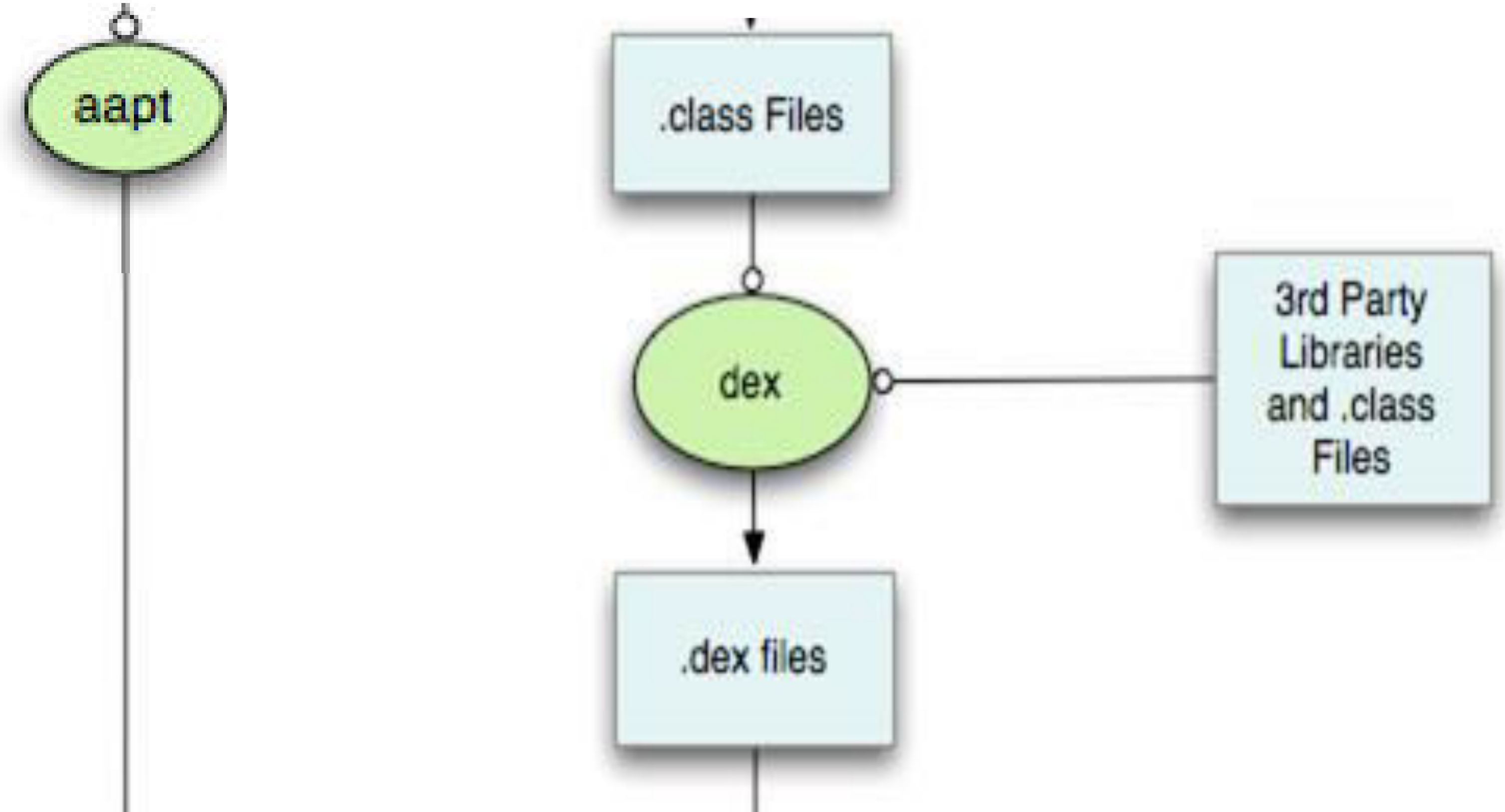
- **The Android Asset Packaging Tool (aapt)** takes your application resource files, such as the *AndroidManifest.xml* file and the XML files for your Activities, and compiles them. *An R.java* is also produced so you can reference your resources from your Java code.

- **The Android Interface Definition Language tool (aidl)** converts any *.aidl* interfaces that you have into Java interfaces.

- **The Java compiler (javac)** compiles all of your Java code, including the *R.java* and *.aidl* files to produce *.class* files as an output output.

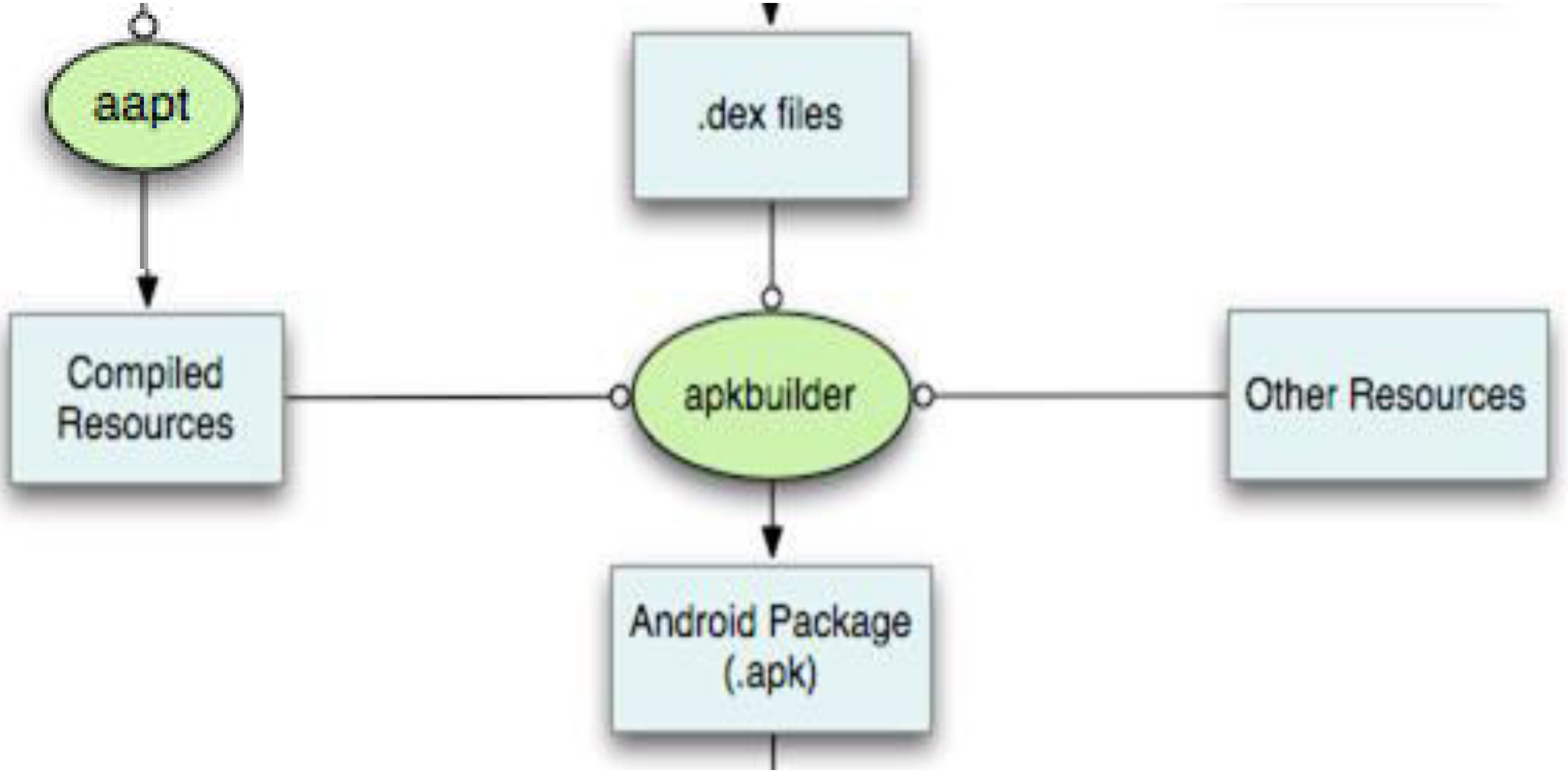
Step-2: Android build system

- **The dex convertor** converts the *.class* files to *Dalvik byte code*. Any 3rd party libraries and *.class* files that you have included in your module build are also converted into *.dex* files so that they can be packaged into the final *.apk* file.



Step-3: Android build system

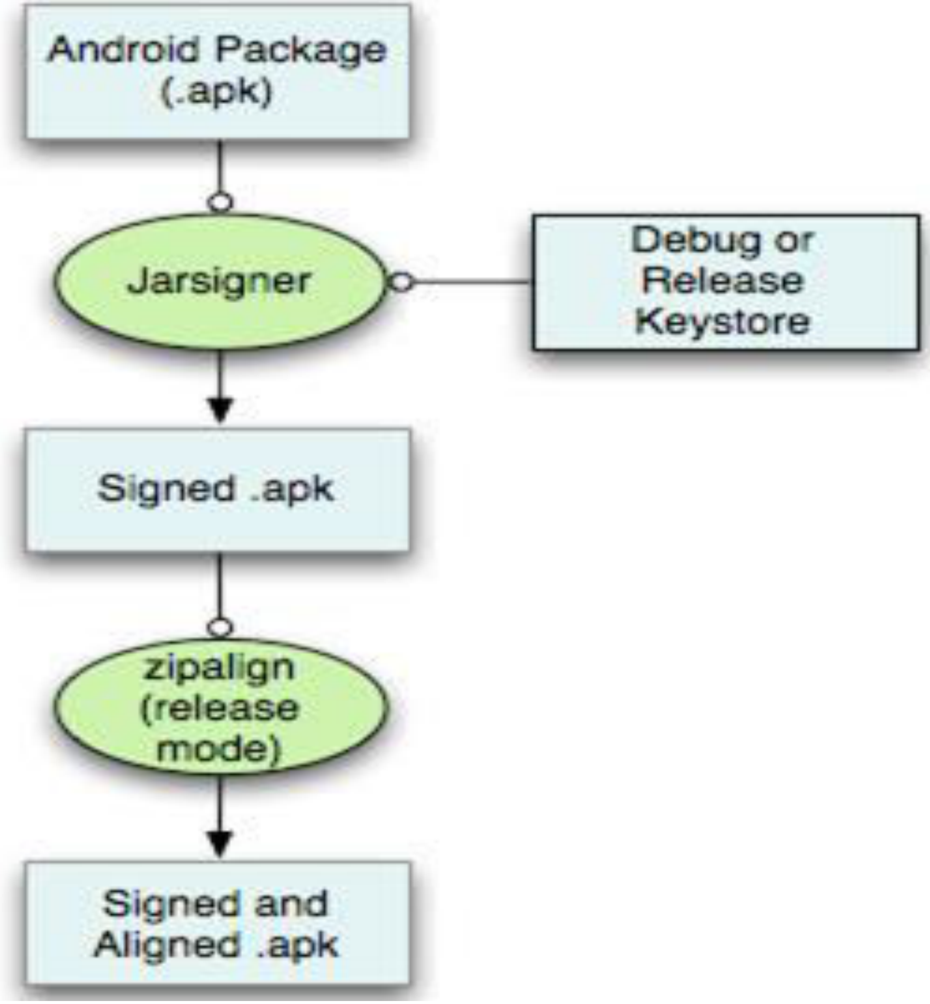
- **The apkbuilder tool** packages all non-compiled resources (such as images), compiled resources, and the .dex files into an *.apk* file.

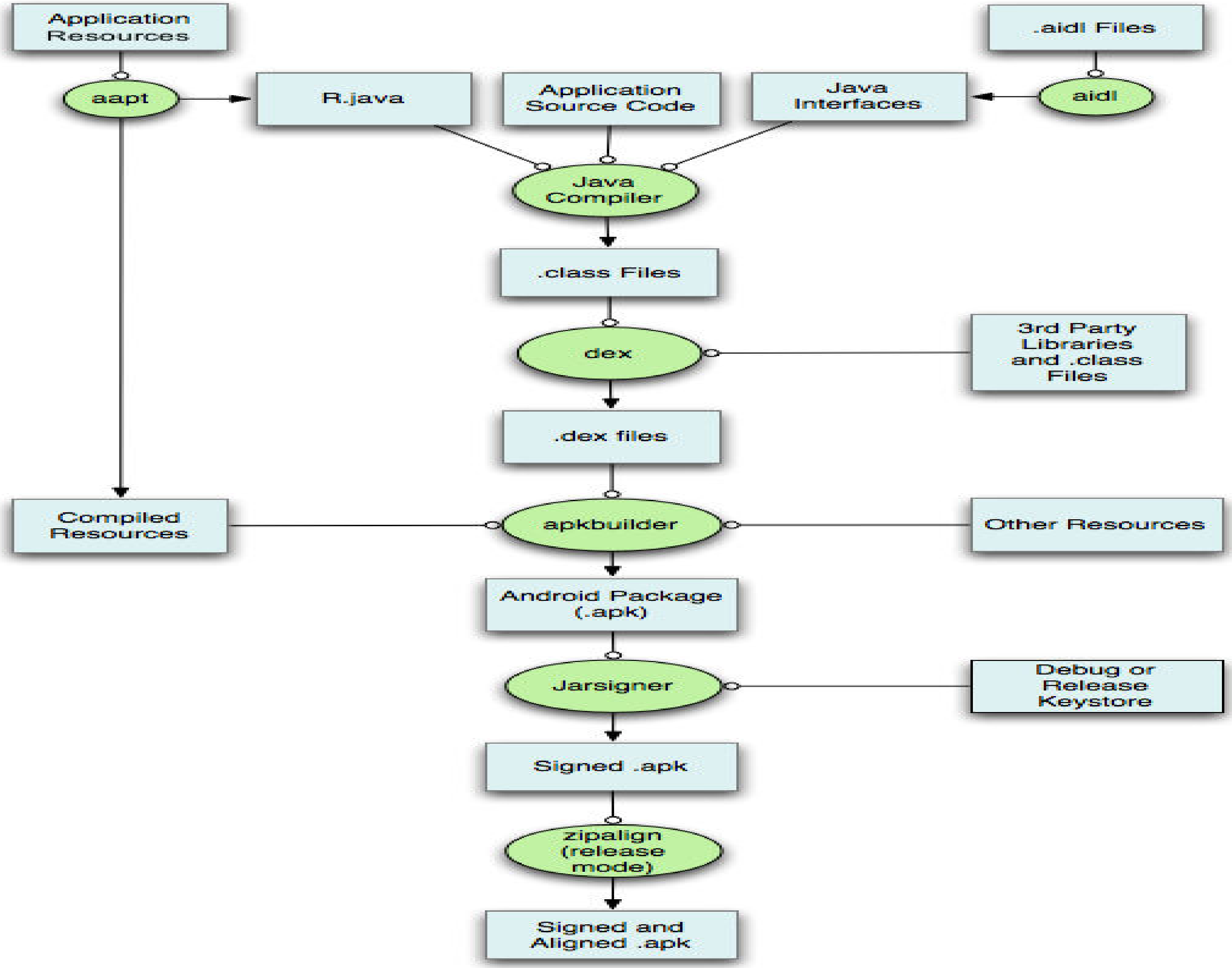


Step-4: Android build system

- **Jarsigner** signs the *.apk* with either a debug or release key before it can be installed to a device.

- **zipalign** is an archive alignment tool that provides important optimization to signed in release mode *.apk* files. Aligning the final *.apk* decreases memory usage when the application is -running on a device.

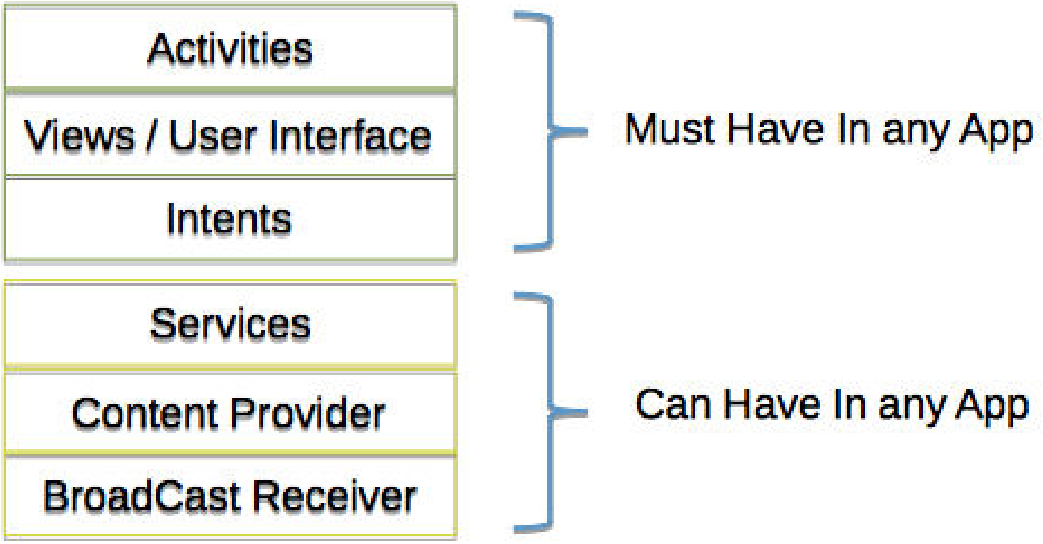



Application Resources
.aidl Files
aapt
R.java
Application Source Code
Java Interfaces
aidl
Java Compiler
.class Files
dex
3rd Party Libraries and .class Files
.dex files
Compiled Resources
apkbuilder
Other Resources
Android Package (.apk)
Jarsigner
Debug or Release Keystore
Signed .apk
zipalign (release mode)
Signed and Aligned .apk

App Components

- An essential building blocks of an Android app. Each component is a different point through which the system can enter your app. Each one is a unique building block that helps define your app's overall behavior.

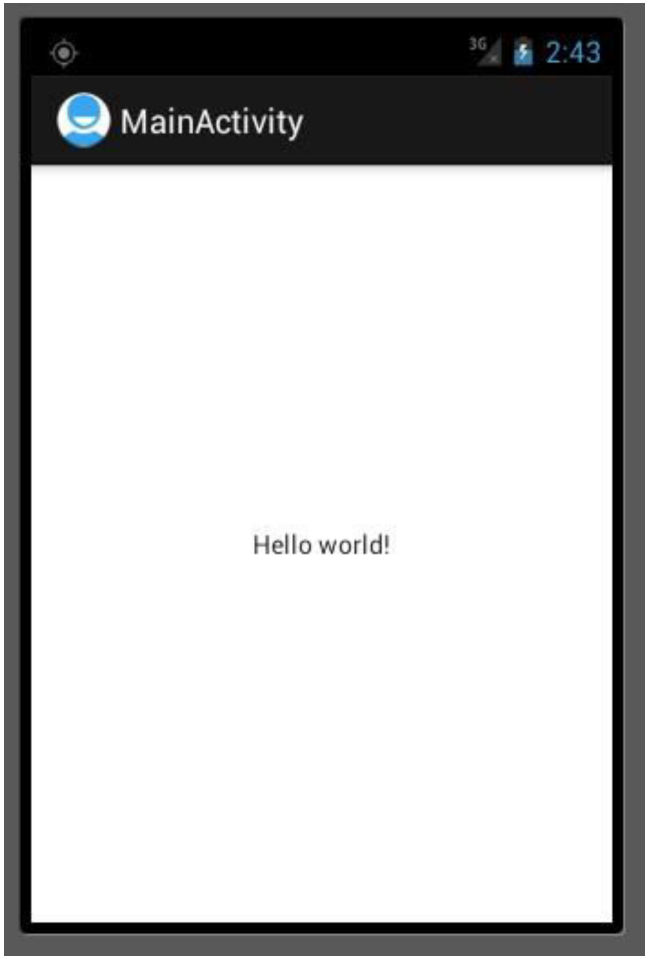
- Several types of app components:
  - **Activities**
  - **Intents**
  - **Services**
  - **Content providers**
  - **BroadCast Receiver**



Activities

- An activity represents a single screen with a user interface.
- **For example:**
  - An email app might have one activity that shows a list of new emails,
  - another activity to compose an email,
  - and another activity for reading emails.

The activities work together to form a cohesive user experience in the email app, each one is independent of the others. As such, a different app can start any one of these activities (if the email app allows it).



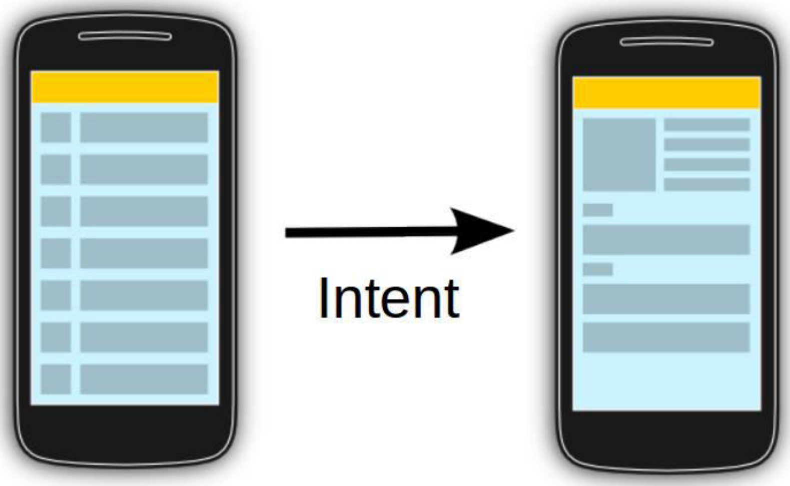
Hello world!

Activities (Cont.)

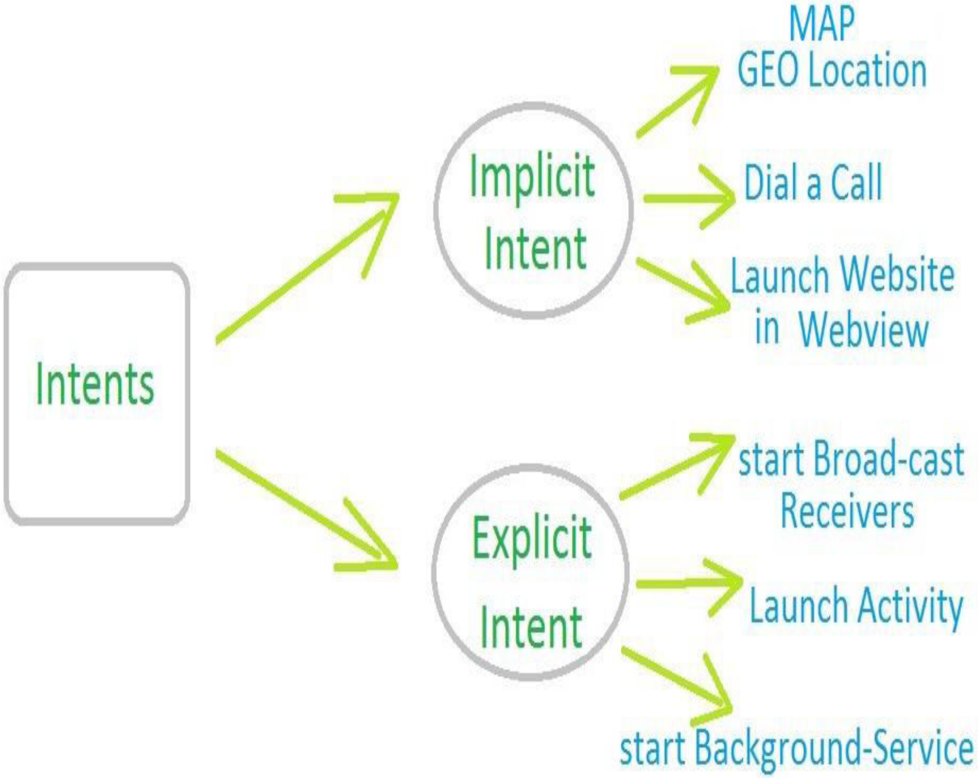
- Several Activities constitute an App

Intents

- An intent is a mechanism for describing a specific action. Briefly describe what should be done.
- In Android, just about everything goes through intents
- **For example:**
  - intent for “send an email.” If your application needs to send mail, you can invoke that intent.
  - if you’re writing a new email application, you can register an activity to handle that intent and replace the standard mail program.
  - The next time somebody tries to send an email, they’ll get the option to use your program instead of the standard one.



types of intents:



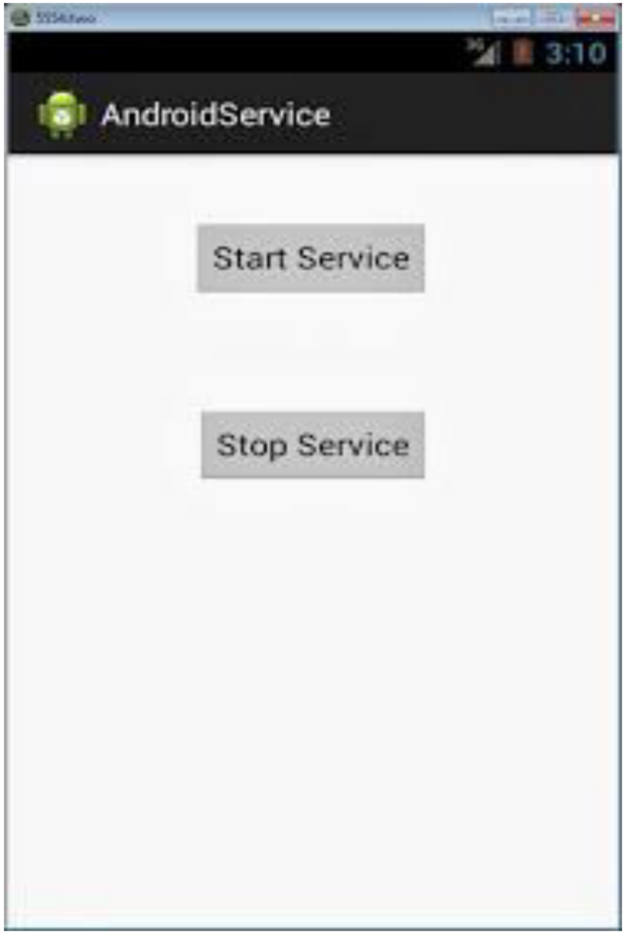
- **Explicit intents**
  - specify the component to start by name (the fully-qualified class name).

    Example:
  - start a new activity in response to a user action or start a service to download a file in the background.
- **Implicit intents**
  - do not name a specific component, but instead declare a general action to perform, which allows a component from another app to handle it.

    Example:
  - if you want to show the user a location on a map, you can use an implicit intent to request that another capable app show a specified location on a map.

Services

- A service is a component that runs in the background:
  - to perform long-running operations
  - to perform work for remote processes.
- A service does not provide a user interface.
- For example:
  - a service might play music in the background while the user is in a different app,
  - or it might fetch data over the network without blocking user interaction with an activity.
  - Another component, such as an activity, can start the service and let it run or bind to it in order to interact with it

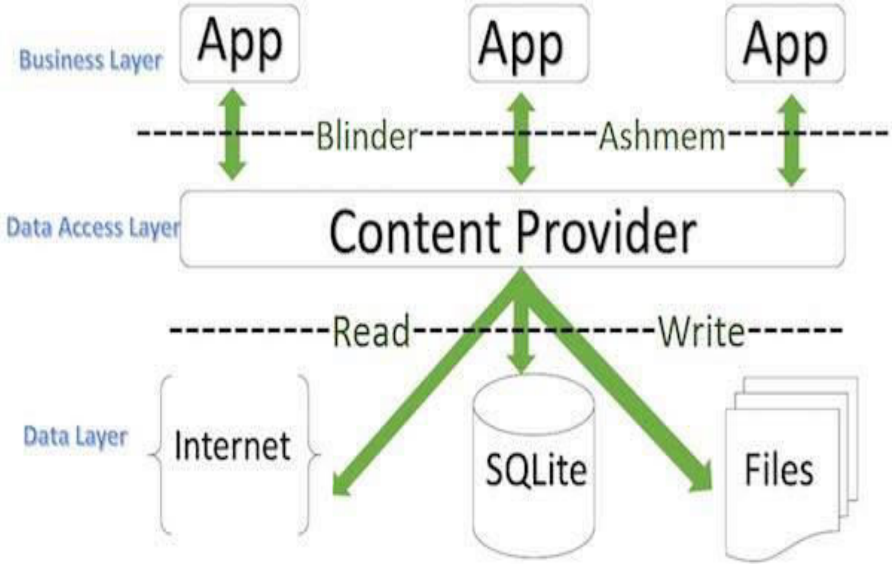


A service can essentially take two forms:

- **Started:** A service is "started"
  - when an application component (such as an activity) starts it by calling *startService().*
  - Usually, a started service performs a single operation and does not return a result to the caller.
  - For example,
    - it might download or upload a file over the network. When the operation is done, the service should stop itself.
- **Bound:** A service is "bound"
  - when an application component binds to it by calling *bindService().*
  - A bound service offers a client-server interface that allows components to interact with the service, send requests, get results, and even do so across processes with interprocess communication (IPC).
  - A bound service runs only as long as another application component is bound to it.

The Content providers.

- Components that manage a shared set of application data.
- The data is stored at one of the following formats:
  1. In the file system.
  2. In an SQLite database.
  3. on the web.
  4. More..



Essentially you can store it in any persistent storage location your application can access.

The Content providers.

- Through the content provider, other applications can query or even modify the data (if the content provider allows it).

- Providers are also useful for reading and writing data that is private to your application and is not shared.

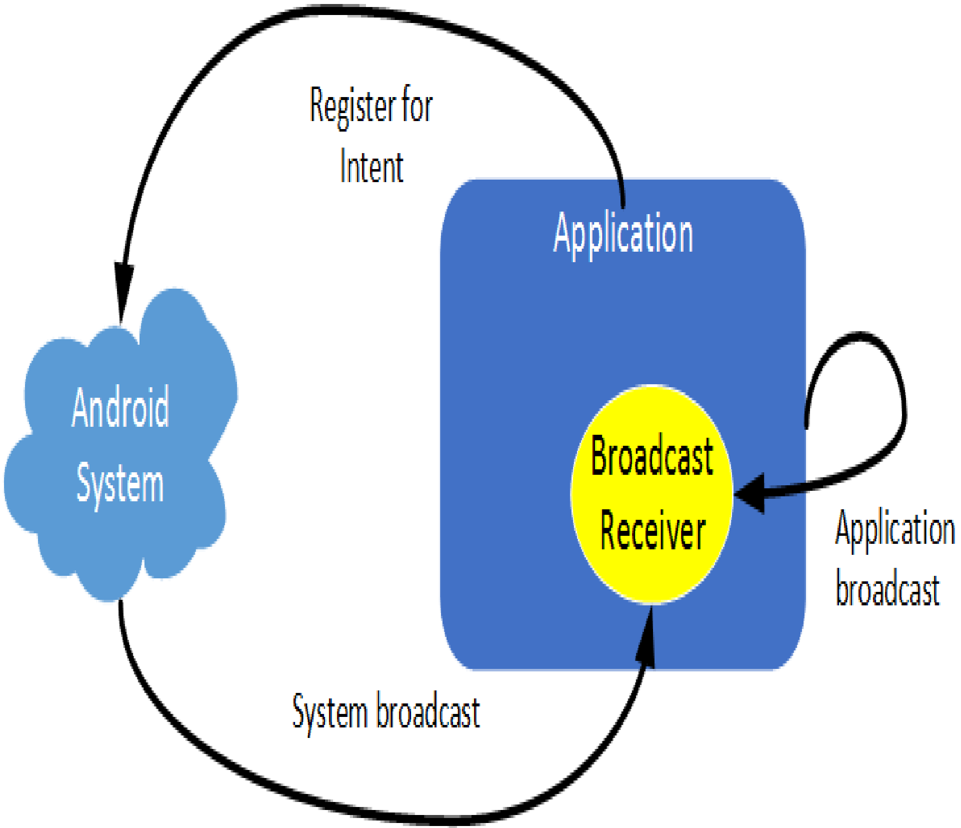
The Content providers – examples.

Here’s a few examples for system content providers:

- any app with the proper permissions can query part of the content provider
  - Contacts.
  - Text messages.
  - Phone calls.

Broadcast receivers

- Broadcast receivers are components in the application that listen for broadcasts and take some action.
- Broadcast Receivers have no user Interface.
- **For example:**
  - When SMS / Call is received
  - Battery low
  - Network state Changed
  - Photo captured from camera
  - Phone Starts



You can find more information about

- **activities**: https://developer.android.com/guide/components/activities.html
- **Intents**
- https://developer.android.com/guide/components/intents-filters.html
- **Services:**
- https://developer.android.com/reference/android/app/Service.html
- **Content providers:**
- https://developer.android.com/reference/android/content/ContentProvider.html
- **Broadcast receivers:**
- http://developer.android.com/reference/android/content/BroadcastReceiver.html