Android Eco-System

Source: http://www.vogella.com/tutorials/Android/article.html#androidstudio_installation

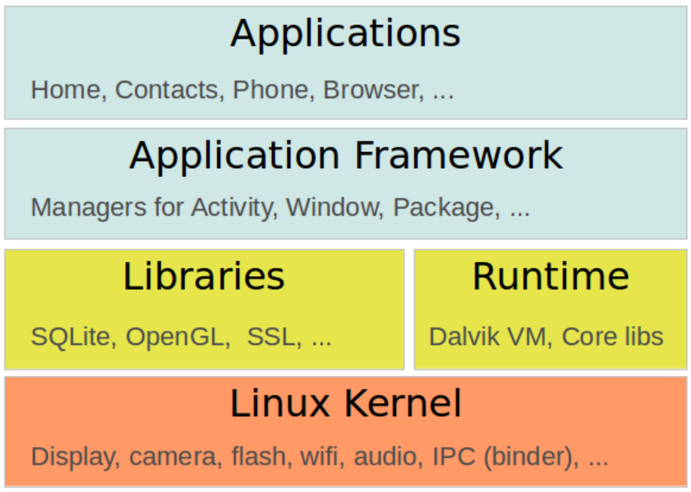
1.1. The Android Operating System (OS)

Android is an operating system **based on the Linux kernel**. The project responsible for developing the Android system is called the ***Android Open Source Project*** **(AOSP) and it lead by Google**.

This Android operating system can be divided into the four areas as depicted in the following graphic. An Android application developer typically works with the two layers on top to create new Android applications.

The levels can be described as:

- **Applications** - The Android Open Source Project contains several default application, like the Browser, Camera, Gallery, Music, Phone and more.
- **Application framework** - An API which allows high-level interactions with the Android system from Android applications.
- **Libraries and runtime** - The libraries for many common framework functions, like, graphic rendering, data storage, web browsing. It also contains the Android Runtime, as well as the core Java libraries for running Android applications.
- **Linux kernel** - Communication layer for the underlying hardware.


Applications
Home, Contacts, Phone, Browser, ...
Application Framework
Managers for Activity, Window, Package, ...
Libraries
SQLite, OpenGL, SSL, ...
Runtime
Dalvik VM, Core libs
Linux Kernel
Display, camera, flash, wifi, audio, IPC (binder), ...

Android is published in different version which is listed in the following table.

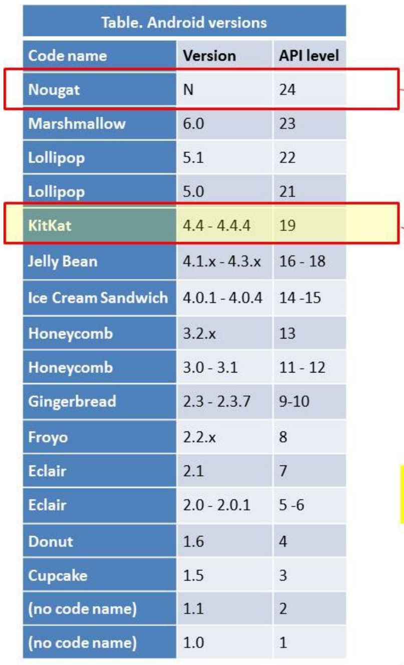
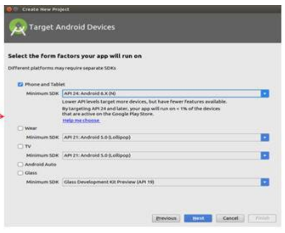


Table. Android versions

Code name	Version	API level
Nougat	N	24
Marshmallow	6.0	23
Lollipop	5.1	22
Lollipop	5.0	21
KitKat	4.4 - 4.4.4	19
Jelly Bean	4.1.x - 4.3.x	16 - 18
Ice Cream Sandwich	4.0.1 - 4.0.4	14 -15
Honeycomb	3.2.x	13
Honeycomb	3.0 - 3.1	11 - 12
Gingerbread	2.3 - 2.3.7	9-10
Froyo	2.2.x	8
Eclair	2.1	7
Eclair	2.0 - 2.0.1	5 -6
Donut	1.6	4
Cupcake	1.5	3
(no code name)	1.1	2
(no code name)	1.0	1

Devices targeted:
<1%



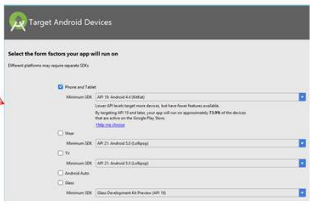
Select the form factors your app will run on

Different platforms may require separate SDKs

☑ Phone and Tablet
Minimum SDK: API 24: Android 6.X (N)
Lower API levels target more devices, but have fewer features available.
By targeting API 24 and later, your app will run on < 1% 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 (API 19)

Devices targeted:
~74%



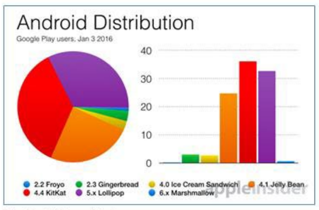
Select the form factors your app will run on

Different platforms may require separate SDKs

☑ Phone and Tablet
Minimum SDK: API 19: Android 4.4 (KitKat)
Lower API levels target more devices, but have fewer features available.
By targeting API 19 and later, your app will run on approximately **73.9%** 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 (API 19)

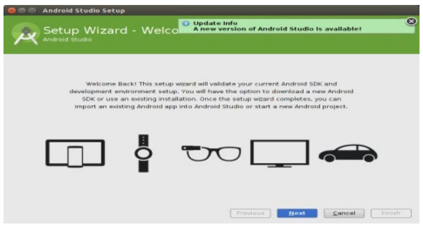
CHECK OTHER ANDROID VERSION & API LEVEL?



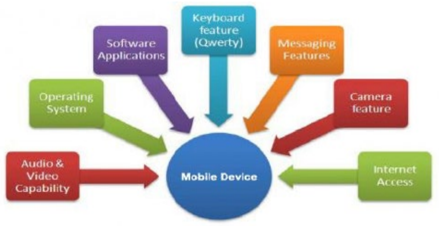
Source: "Android development with Android Studio", at:
http://www.vogella.com/tutorials/Android/article.html#androidstudio_installation

1.2. How to develop Android applications

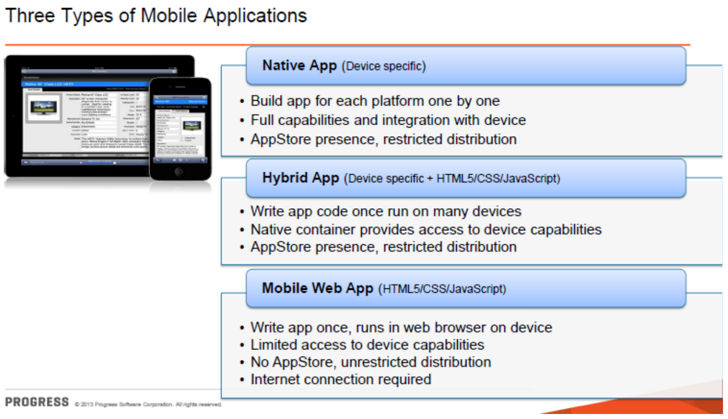
Android applications are primarily written in the **Java programming language**. During development the developer creates the Android specific configuration files and writes the application logic in the Java programming language.



Welcome Back! This setup wizard will validate your current Android SDK and development environment setup. You will have the option to download a new Android SDK or use an existing installation. Once the setup wizard completes, you can import an existing Android app into Android Studio or start a new Android project.



The Android development tooling converts these application files into an Android application. If the developer triggers the deployment, **the whole Android application is compiled, packaged, deployed and potentially started**.



Three Types of Mobile Applications

Native App (Device specific)

- Build app for each platform one by one
- Full capabilities and integration with device
- AppStore presence, restricted distribution

Hybrid App (Device specific + HTML5/CSS/JavaScript)

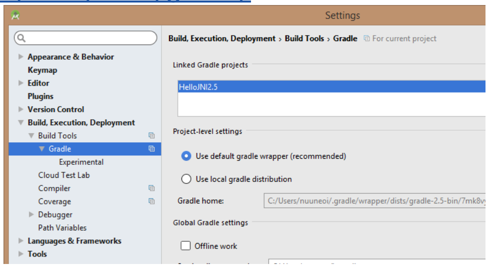
- Write app code once run on many devices
- Native container provides access to device capabilities
- AppStore presence, restricted distribution

Mobile Web App (HTML5/CSS/JavaScript)

- Write app once, runs in web browser on device
- Limited access to device capabilities
- No AppStore, unrestricted distribution
- Internet connection required

PROGRESS © 2013 Progress Software Corporation. All rights reserved.

The *Android Software Development Kit* (Android SDK) and the **Gradle tooling** contain the necessary tools to create, compile and package Android applications. The Android team provides a **Gradle plug-in** to build Android applications. You find the available versions of this plug-in under the following URL: https://jcenter.bintray.com/com/android/tools/build/gradle/



Build, Execution, Deployment › Build Tools › Gradle   For current project

Linked Gradle projects: HelloJNI2.5

Project-level settings

[✓ Use default gradle wrapper (recommended)]

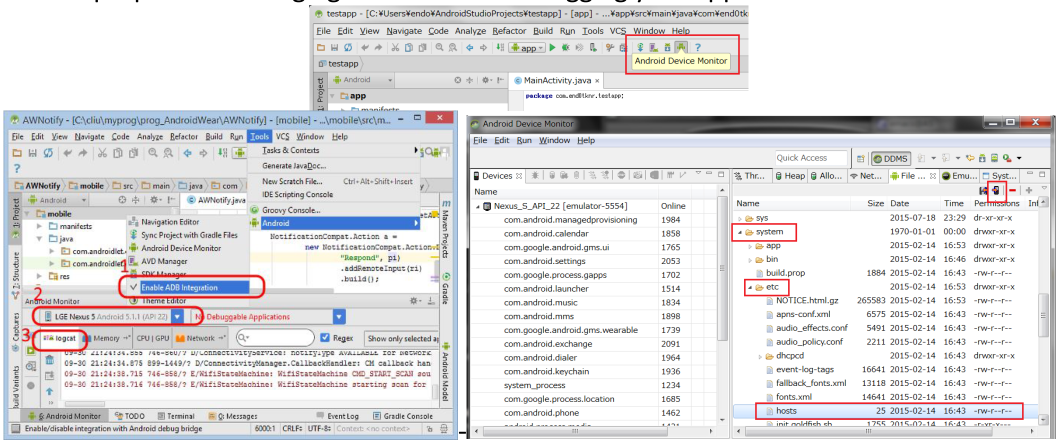
Use local gradle distribution

Gradle home: C:/Users/nuuneoi/.gradle/wrapper/dists/gradle-2.5-bin/7mk8v

Global Gradle settings

Offline work

The Android SDK contains the ***Android debug bridge* (ADB).** ADB is a tool that allows you to connect to a virtual or real Android device, for the purpose of managing the device or debugging your application.


testapp - [C:¥Users¥endo¥AndroidStudioProjects¥testapp] - [app] - ...¥app¥src¥main¥java¥com¥end0tk
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
app
Android Device Monitor
MainActivity.java
AWNotify - [C:\cliu\myprog\prog_AndroidWear\AWNotify] - [mobile] - ...\mobile\src\m...
Tasks & Contexts
Generate JavaDoc...
New Scratch File... Ctrl+Alt+Shift+Insert
IDE Scripting Console
Groovy Console...
Android
Navigation Editor
Sync Project with Gradle Files
Android Device Monitor
AVD Manager
SDK Manager
Enable ADB Integration
Theme Editor
1
2
3
Android Monitor
LGE Nexus 5 Android 5.1.1 (API 22)
No Debuggable Applications
logcat
Memory
CPU | GPU
Network
Regex
Show only selected a
Android Monitor
TODO
Terminal
0: Messages
Event Log
Gradle Console
Enable/disable integration with Android debug bridge
Android Device Monitor
File Edit Run Window Help
Quick Access
DDMS
Devices
Thr...
Heap
Allo...
Net...
File ...
Emu...
Syst...

```
package com.end0tknr.testapp;
```

```
NotificationCompat.Action a =
        new NotificationCompat.Action.
                "Respond", pi)
                .addRemoteInput(ri)
                .build();
```

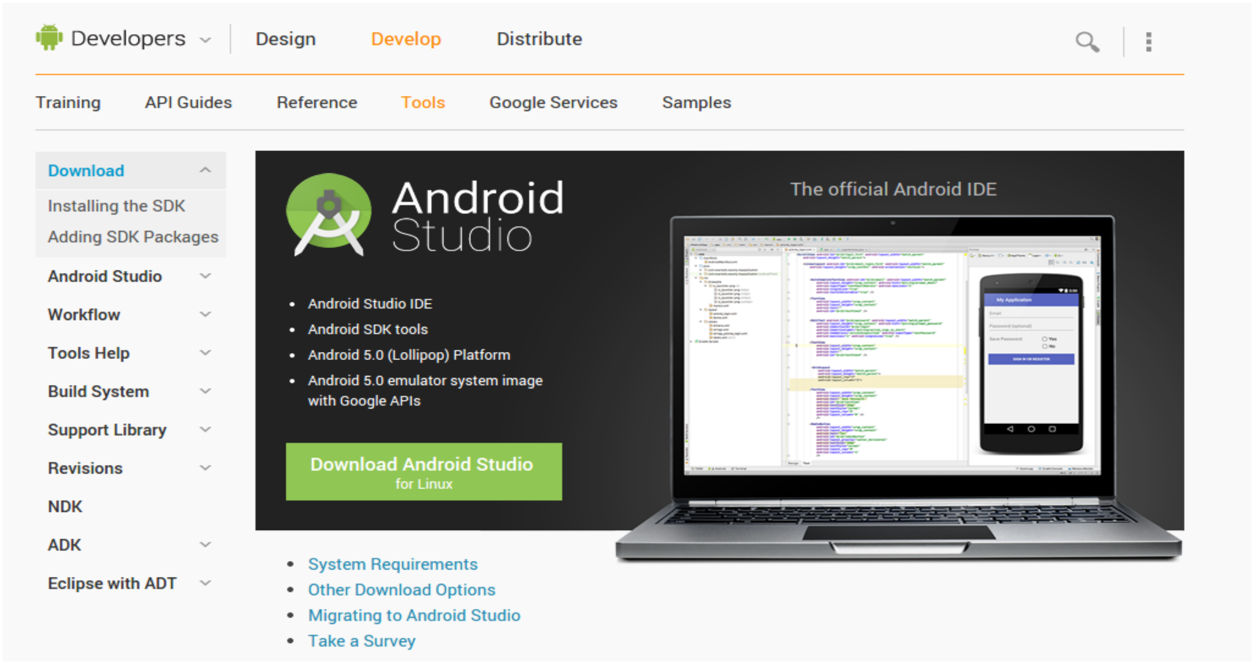
```
09-30 21:24:34.855 746-860/? D/ConnectivityService: notifyType AVAILABLE for Network
09-30 21:24:34.875 899-1449/? D/ConnectivityManager.CallbackHandler: CM callback han
09-30 21:24:38.715 746-858/? E/WifiStateMachine: WifiStateMachine CMD_START_SCAN sou
09-30 21:24:38.716 746-858/? E/WifiStateMachine: WifiStateMachine starting scan for
```

Name	
Nexus_S_API_22 [emulator-5554]	Online
com.android.managedprovisioning	1984
com.android.calendar	1858
com.google.android.gms.ui	1765
com.android.settings	2053
com.google.process.gapps	1702
com.android.launcher	1514
com.android.music	1834
com.android.mms	1898
com.google.android.gms.wearable	1739
com.android.exchange	2091
com.android.dialer	1964
com.android.keychain	1936
system_process	1234
com.google.process.location	1685
com.android.phone	1462

Name	Size	Date	Time	Permissions
sys		2015-07-18	23:29	dr-xr-xr-x
system		1970-01-01	00:00	drwxr-xr-x
app		2015-02-14	16:53	drwxr-xr-x
bin		2015-02-14	16:46	drwxr-xr-x
build.prop	1884	2015-02-14	16:43	-rw-r--r--
etc		2015-02-14	16:53	drwxr-xr-x
NOTICE.html.gz	265583	2015-02-14	16:53	-rw-r--r--
apns-conf.xml	6575	2015-02-14	16:43	-rw-r--r--
audio_effects.conf	5491	2015-02-14	16:43	-rw-r--r--
audio_policy.conf	2211	2015-02-14	16:43	-rw-r--r--
dhcpcd		2015-02-14	16:43	drwxr-xr-x
event-log-tags	16641	2015-02-14	16:43	-rw-r--r--
fallback_fonts.xml	13118	2015-02-14	16:43	-rw-r--r--
fonts.xml	14641	2015-02-14	16:43	-rw-r--r--
hosts	25	2015-02-14	16:43	-rw-r--r--

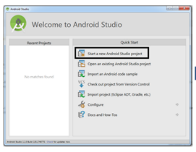
1.3. Android Developer Tools: Android Studio

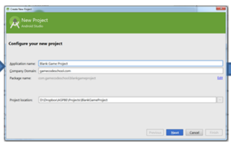
Google provides an IDE called *Android Studio* as the preferred development environment for creating Android applications. This IDE is based on the **IntelliJ IDE.**

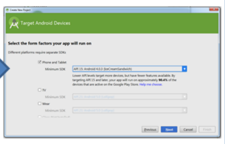


The Android tools provide specialized editors for Android specific files. Most of Android's configuration files are based on XML. In this case these editors allow you to switch between the XML representation of the file and a structured user interface for entering the data.

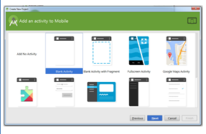
Create a new Android project

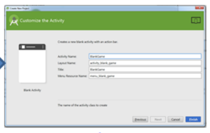

Start a new Android Studio Project


Configure your new project dialog box

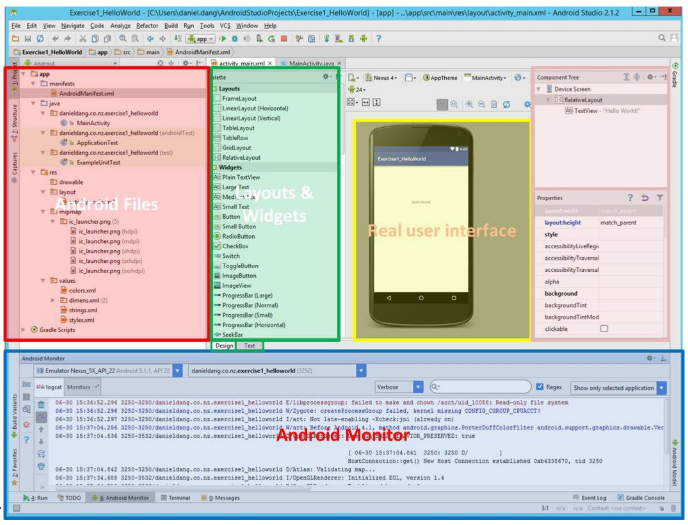

Target Android Devices screen

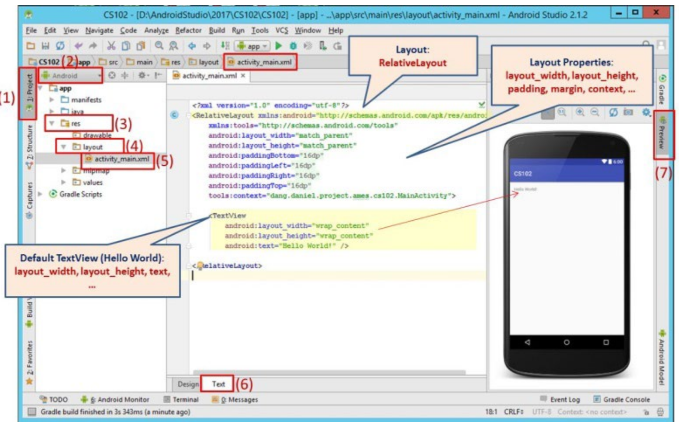
- Make sure the Phones and Tablets checkbox has a tick in it. It probably has by default.
- For the Minimum SDK to target option, a good rule of thumb is to pick a newer version of Android that does not exclude too many older devices. At the time of writing, **the API 19: Android 4.4 (KitKat)** is a good choice. But it doesn't matter too much while we are learning. If you are unsure just leave it at the default that Android Studio selects.


Add an Activity to mobile


Customize the Activity

_Application name: **HelloWorld**
_Company Domain: **nz.ac.ames.project**
_Target Android devices: **Phone and Tablet;**
+ *Minimum SDK: **API 19: Android 4.4 (KitKat)***
_Add an Activity to Mobile: **Empty Activity;**
+ *Activity name: **MainActivity***
+ *Layout name: **activity_main***

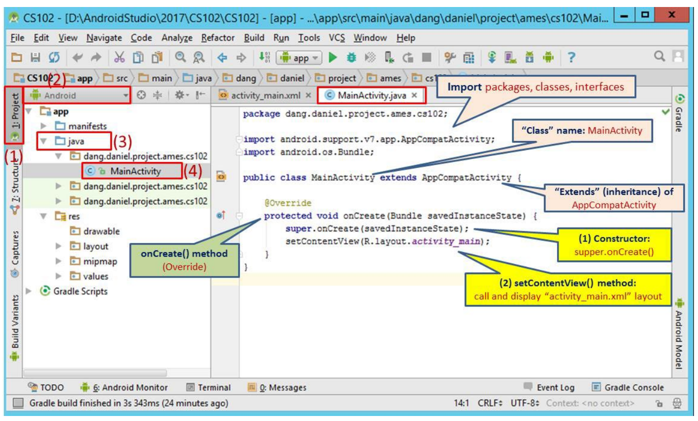

Exercise1_HelloWorld - [C:\Users\daniel.dang\AndroidStudioProjects\Exercise1_HelloWorld] - [app] - ..\app\src\main\res\layout\activity_main.xml - Android Studio 2.1.2
Android Files
Layouts & Widgets
Real user interface
Android Monitor


CS102 - [D:\AndroidStudio\2017\CS102\CS102] - [app] - ..\app\src\main\res\layout\activity_main.xml - Android Studio 2.1.2

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/andro
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="16dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    tools:context="dang.daniel.project.ames.cs102.MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!" />

</RelativeLayout>
```

(1)
(2)
(3)
(4)
(5)
(6)
(7)
Layout: RelativeLayout
Layout Properties: layout_width, layout_height, padding, margin, context, ...
Default TextView (Hello World): layout_width, layout_height, text, ...
Gradle build finished in 3s 343ms (a minute ago)

Activity_main.xml

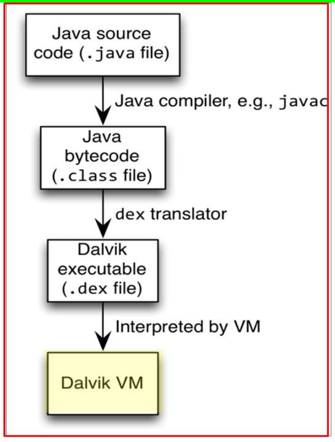

CS102 - [D:\AndroidStudio\2017\CS102\CS102] - [app] - ..\app\src\main\java\dang\daniel\project\ames\cs102\Mai...

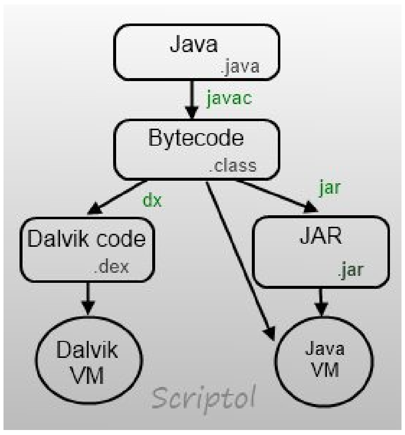
```
package dang.daniel.project.ames.cs102;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

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

(1)
(2)
(3)
(4)
Import packages, classes, interfaces
"Class" name: MainActivity
"Extends" (inheritance) of AppCompatActivity
onCreate() method (Override)
(1) Constructor: supper.onCreate()
(2) setContentView() method: call and display "activity_main.xml" layout
Gradle build finished in 3s 343ms (24 minutes ago)

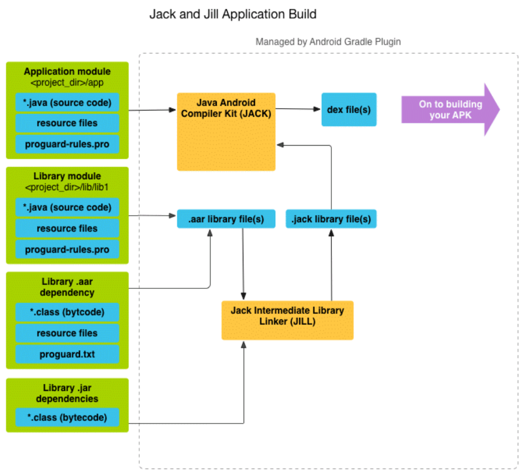
MainActivity.java

1.4. Conversion process from source code to Android application

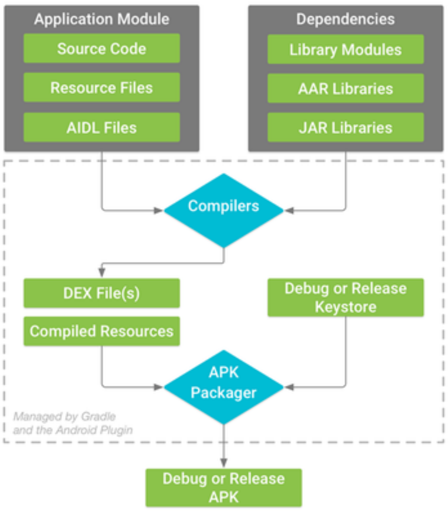

Java source code (.java file)
Java compiler, e.g., javac
Java bytecode (.class file)
dex translator
Dalvik executable (.dex file)
Interpreted by VM
Dalvik VM
Application compilation process



Dalvik vs. Java ME



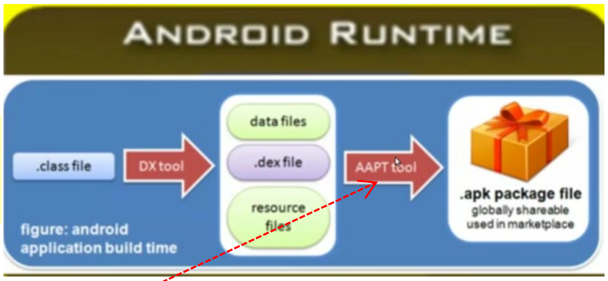
The **Java source files** are converted to **Java class files** by the ***Java compiler***. The Android SDK contains a tool called ***dex translator*** which converts **Java class files** into a ***.dex* (Dalvik Executable) file**. All class files of the application are placed in this *.dex* file. During this conversion process redundant information in the class files are optimized in the *.dex* file. For example, if the same String is found in different class files, the *.dex* file contains only one reference of this String. These *.dex* files are therefore much smaller in size than the corresponding class files.



The "build" process of a typical Android app module:

1. **The compilers convert your source code into DEX (Dalvik Executable) files;**
2. **The APK Packager combines the DEX files and compiled resources into a single APK (Android Application Package);**
3. **The APK Packager signs your APK using either the debug or release keystore;**

At the end of the build process, you have either a **debug APK or release APK of your app** that you can use to deploy, test, or release to external users.



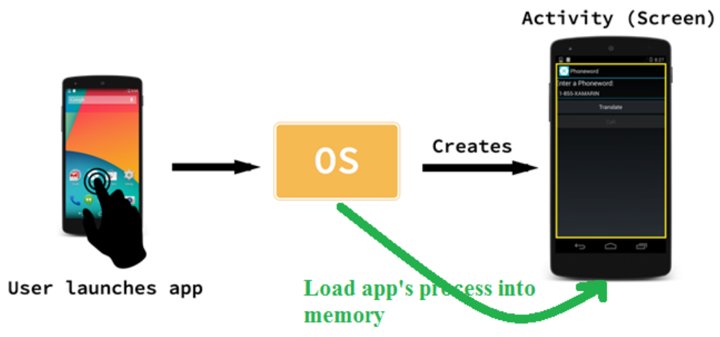
The ***.dex* file** and **the resources** of an Android project, e.g., the images and XML files, are packed into an ***.apk* (Android Package) file**. The program *aapt* (Android Asset Packaging Tool) performs this step. The resulting *.apk* file contains all necessary data to run the Android application and can be deployed to an Android device via the ***ADB* tool**.
As of Android 5.0, the ***Android RunTime* (ART)** is used as runtime for all Android applications. ART uses *Ahead Of Time* compilation. During the installation of an application on an Android device, the application code is translated into machine code.

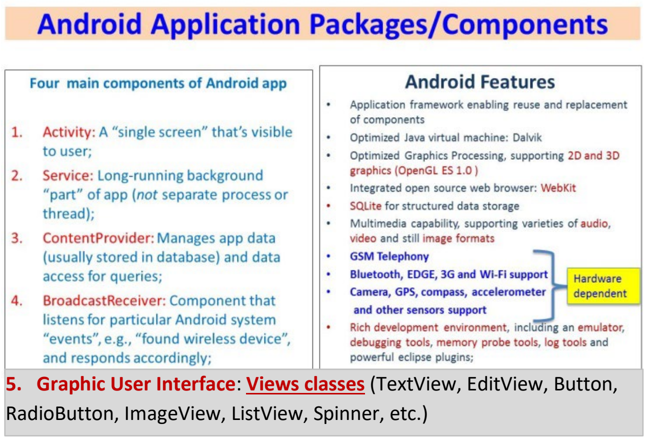
- This results in approx. 30% larger compile code, but allows faster execution at startup.
- This also saves battery life, as the compilation is only done once, during the first start of the application.

The **dex2oat tool** takes the **.dex file** created by the Android tool change and compiles that into an **Executable and Linkable Format (ELF file)**. This file contains the **dex code**, **compiled native code** and **meta-data**. Keeping the .dex code allows that existing tools still work.

The **garbage collection in ART** has been optimized to reduce times in which the application freezes.

1.5. Android Application Packages & Components





Android Application Packages/Components

Four main components of Android app

1. Activity: A "single screen" that's visible to user;
2. Service: Long-running background "part" of app (*not* separate process or thread);
3. ContentProvider: Manages app data (usually stored in database) and data access for queries;
4. BroadcastReceiver: Component that listens for particular Android system "events", e.g., "found wireless device", and responds accordingly;

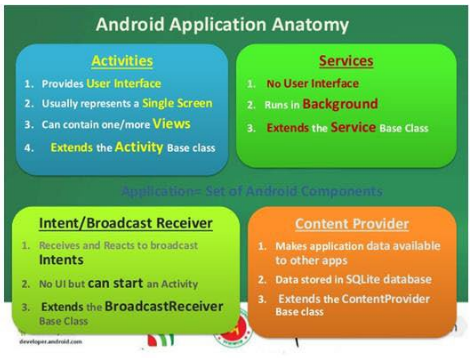
Android Features

- Application framework enabling reuse and replacement of components
- Optimized Java virtual machine: Dalvik
- Optimized Graphics Processing, supporting 2D and 3D graphics (OpenGL ES 1.0)
- Integrated open source web browser: WebKit
- SQLite for structured data storage
- Multimedia capability, supporting varieties of audio, video and still image formats
- **GSM Telephony**
- **Bluetooth, EDGE, 3G and Wi-Fi support**
- **Camera, GPS, compass, accelerometer and other sensors support**

Hardware dependent

- Rich development environment, including an emulator, debugging tools, memory probe tools, log tools and powerful eclipse plugins;

5. Graphic User Interface: **Views classes** (TextView, EditView, Button, RadioButton, ImageView, ListView, Spinner, etc.)



Android Application Anatomy

Activities
1. Provides User Interface
2. Usually represents a Single Screen
3. Can contain one/more Views
4. Extends the Activity Base class

Services
1. No User Interface
2. Runs in Background
3. Extends the Service Base Class

Application= Set of Android Components

Intent/Broadcast Receiver
1. Receives and Reacts to broadcast Intents
2. No UI but can start an Activity
3. Extends the BroadcastReceiver Base Class

Content Provider
1. Makes application data available to other apps
2. Data stored in SQLite database
3. Extends the ContentProvider Base class

developer.android.com

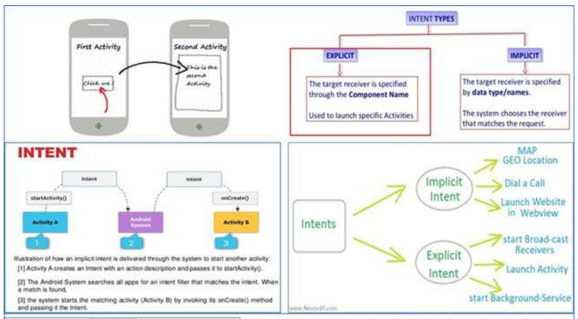
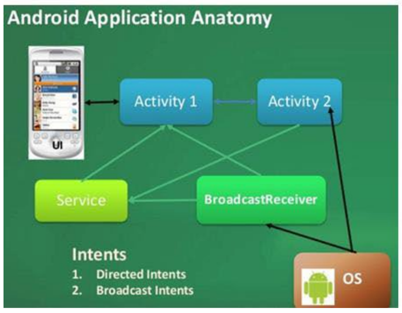

First Activity
Click me!
Second Activity
This is the second Activity
INTENT TYPES
EXPLICIT
The target receiver is specified through the Component Name
Used to launch specific Activities
IMPLICIT
The target receiver is specified by data type/names.
The system chooses the receiver that matches the request.
INTENT
Intent
startActivity()
Activity A
Android System
onCreate()
Activity B
Intents
Implicit Intent
MAP GEO Location
Dial a Call
Launch Website in Webview
Explicit Intent
start Broad-cast Receivers
Launch Activity
start Background-Service

Illustration of how an implicit intent is delivered through the system to start another activity:
[1] Activity A creates an Intent with an action description and passes it to startActivity().
[2] The Android System searches all apps for an intent filter that matches the intent. When a match is found,
[3] the system starts the matching activity (Activity B) by invoking its onCreate() method and passing it the Intent.

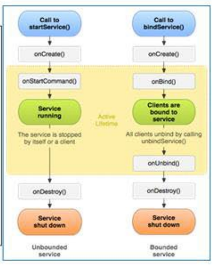


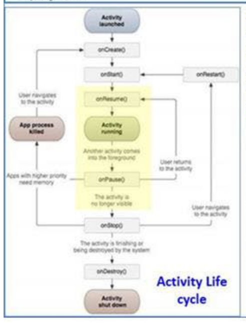
Service

A **service** is a component that runs in the background to **perform long-running operations without needing to interact with the user** and it works even if application is destroyed.

A Foreground Service in android is a background service which keeps running even after the parent application is closed. Popular use **cases:**

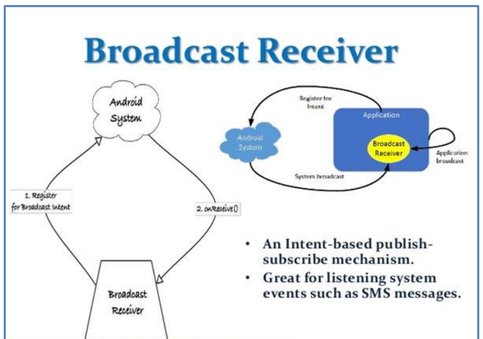
- For downloading a file in background from a server
- For playing songs in background in a music player app
- For showing the status of Connection to the chat server for a chat messenger app




Activity launched
onCreate()
onStart()
onRestart()
onResume()
Activity running
App process killed
Another activity comes into the foreground
onPause()
User returns to the activity
Apps with higher priority need memory
The activity is no longer visible
onStop()
User navigates to the activity
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down
Activity Life cycle

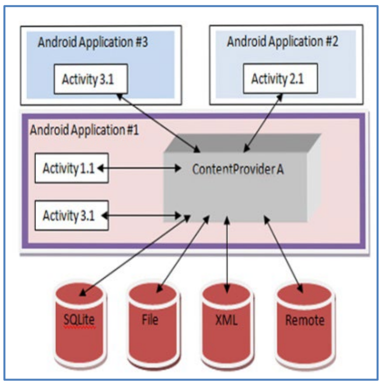
An Activity is an application component that provides a screen with which users can interact in order to do something, such as dial the phone, take a photo, send an email, or view a map. Each activity is given a window in which to draw its user interface.

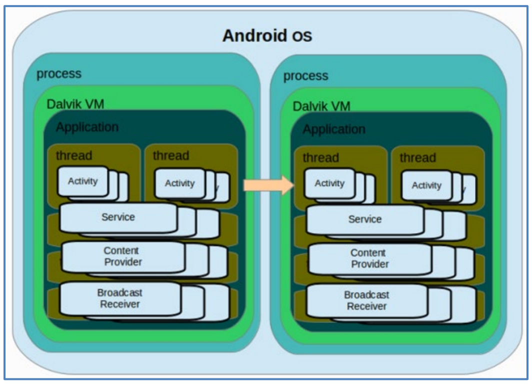
Callback	Description
onCreate()	This is the first callback and called when the activity is first created.
onStart()	This callback is called when the activity becomes visible to the user.
onResume()	This is called when the user starts interacting with the application.
onPause()	The paused activity does not receive user input and cannot execute any code and called when the current activity is being paused and the previous activity is being resumed.
onStop()	This callback is called when the activity is no longer visible.
onDestroy()	This callback is called before the activity is destroyed by the system.
onRestart()	This callback is called when the activity restarts after stopping it.



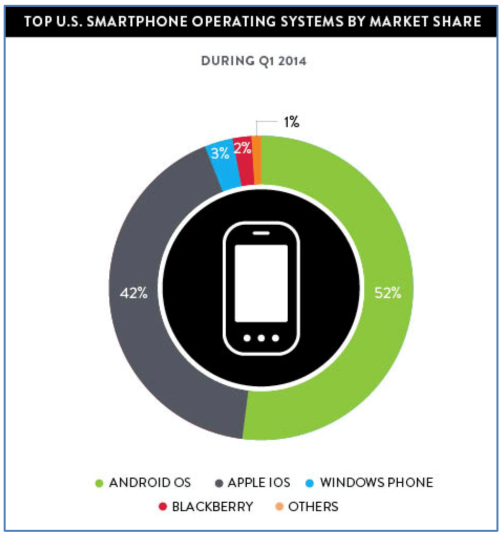
- An Intent-based publish-subscribe mechanism.
- Great for listening system events such as SMS messages.

☐ Courtesy to Everything of Android, slideshare.

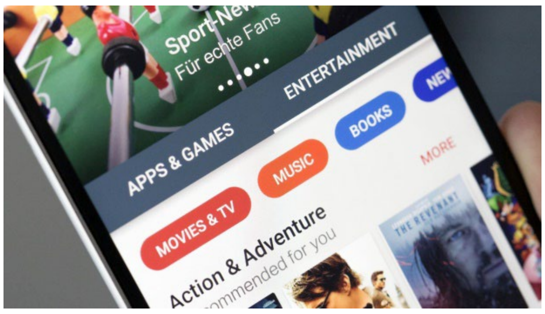




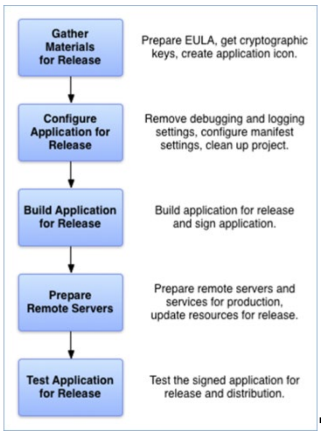
1.6. Google Play

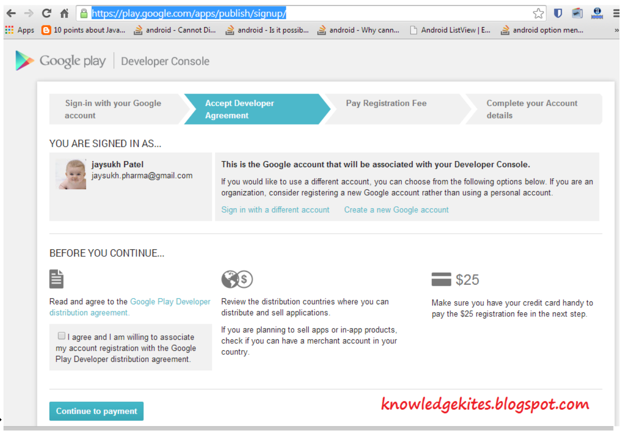


Google offers the *Google Play* service, a marketplace in which programmers can offer their application to Android users. Customers use the *Google Play* application which allows them to buy and install applications from the Google Play service.



Google Play also offers an update service. If an application developer uploads a new version of his application to Google Play, this service notifies existing users about the update allows them to install the update.
Google Play provides access to services and libraries for Android application programmers, too. For example, it provides a service to use and display Google Maps. Providing these services via Google Play has the advantage that they are available for older Android releases. Google can update them without the need for an update of the Android release on the phone.


Gather Materials for Release
Prepare EULA, get cryptographic keys, create application icon.
Configure Application for Release
Remove debugging and logging settings, configure manifest settings, clean up project.
Build Application for Release
Build application for release and sign application.
Prepare Remote Servers
Prepare remote servers and services for production, update resources for release.
Test Application for Release
Test the signed application for release and distribution.



https://play.google.com/apps/publish/signup/

Google play | Developer Console

Sign-in with your Google account | Accept Developer Agreement | Pay Registration Fee | Complete your Account details

YOU ARE SIGNED IN AS...

jaysukh Patel
jaysukh.pharma@gmail.com

This is the Google account that will be associated with your Developer Console.
If you would like to use a different account, you can choose from the following options below. If you are an organization, consider registering a new Google account rather than using a personal account.
Sign in with a different account   Create a new Google account

BEFORE YOU CONTINUE...

Read and agree to the Google Play Developer distribution agreement.
☐ I agree and I am willing to associate my account registration with the Google Play Developer distribution agreement.

Review the distribution countries where you can distribute and sell applications.
If you are planning to sell apps or in-app products, check if you can have a merchant account in your country.

$25
Make sure you have your credit card handy to pay the $25 registration fee in the next step.

Continue to payment

knowledgekites.blogspot.com

Any apps which violate the terms and conditions will be suspended from the store and repeated violations can result in the termination of your developer account. That probably sounds worse than it actually is. If you are creating original and well behaved apps then you don't really have to worry.