Introduction

Coventry University

- Born 1995
- Created app called **Trimit** in 2011
- Sold to Yahoo for $30 million US dollars in 2013

http://en.wikipedia.org/wiki/Nick_D'Aloisio

LAB 1

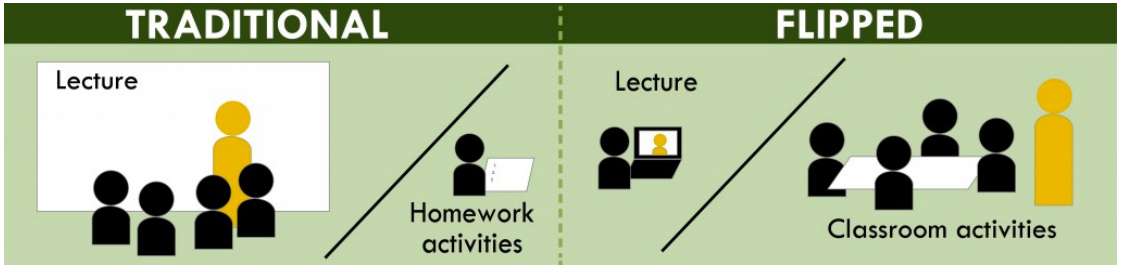
- Module structure
- Why mobile apps?
- First bite on Android
- Tools and resources
- Setting up IDE

Weekly Lab Format

- 3 labs each week
- 2 with detailed instruction, last to explore/work on assignment
- All material on Github (https://github.com/covcom/300CEM), or simply google ‘300CEM Github’

Weekly Lab Format

- 30' • Previous worksheet
- 30' • New worksheet
- 30' • Presentation by tutors on concepts and theory
- 30+' • Continue on new worksheet



http://www.washington.edu/teaching/files/2012/11/FlippedClassroomUWcolors-1-1024x242.jpg

Module structure

Requirements

- Existing knowledge and programming experience

Outcomes

- Demonstrate familiarity with the **Java** Programming language and the Android Studio.
- **Design** applications suitable for Android devices.
- Use the Android **software development kit** and emulator to develop applications for the Android platform.
- Make use of the main modes of **interaction** available on a smartphone platform.

Module structure

1. Android Studio
2. The Java language
3. XML and gradle
4. Simple views and controls
5. Composit views
6. Data persistence
7. Testing
8. Wearables
9. Graphics and animation
10. Location services and Maps
11. Multimedia

Module structure – assessment

- Deadline: 13 December 2015
- Assignment sheet on Moodle!

Why mobile apps?

A mobile app is a computer program designed to run on smartphones, tablet computers and other mobile devices.

http://en.wikipedia.org/wiki/Mobile_app

Why mobile apps?

Transformative Devices

- Ubiquitous and compact
- long battery life
- Environment-aware
  - location (coarse/fine), angle, acceleration
- Multi-channel communication
  - cellular, Wi-Fi, Bluetooth, voice, data

Why mobile apps?

Reaching Customers

- Disruptive technology
- Personalized devices
- Always online
- Packed with advanced hardware
  - Immediacy
  - Short bursts
  - Not tied to ‘business hours’

Why mobile apps?

Changing Business Processes

- Processes fit current technology
- As technology changes, processes can be radically transformed
- Ubiquitous personal technology
  - Always on connectivity
  - BYOD
  - Advanced hardware (sensors, etc.)

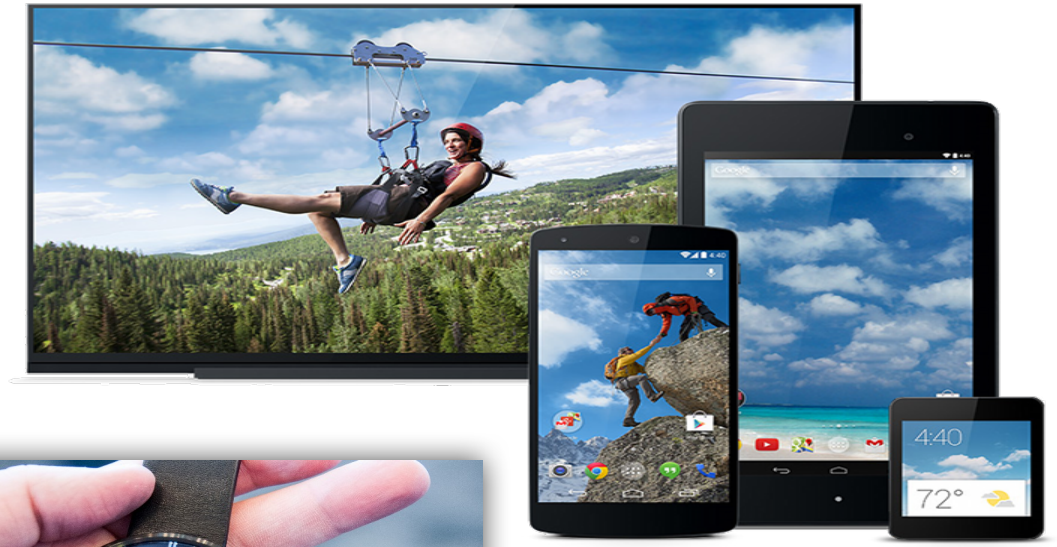
Why mobile apps?

Making Money

- App Store
  - distribution, returns and payment
- Monetisation
  - one-off fee, in-app advertising (view/click), subscription, in-app purchase

First bite on Android

- Complete
- Open
- Free



First bite on Android

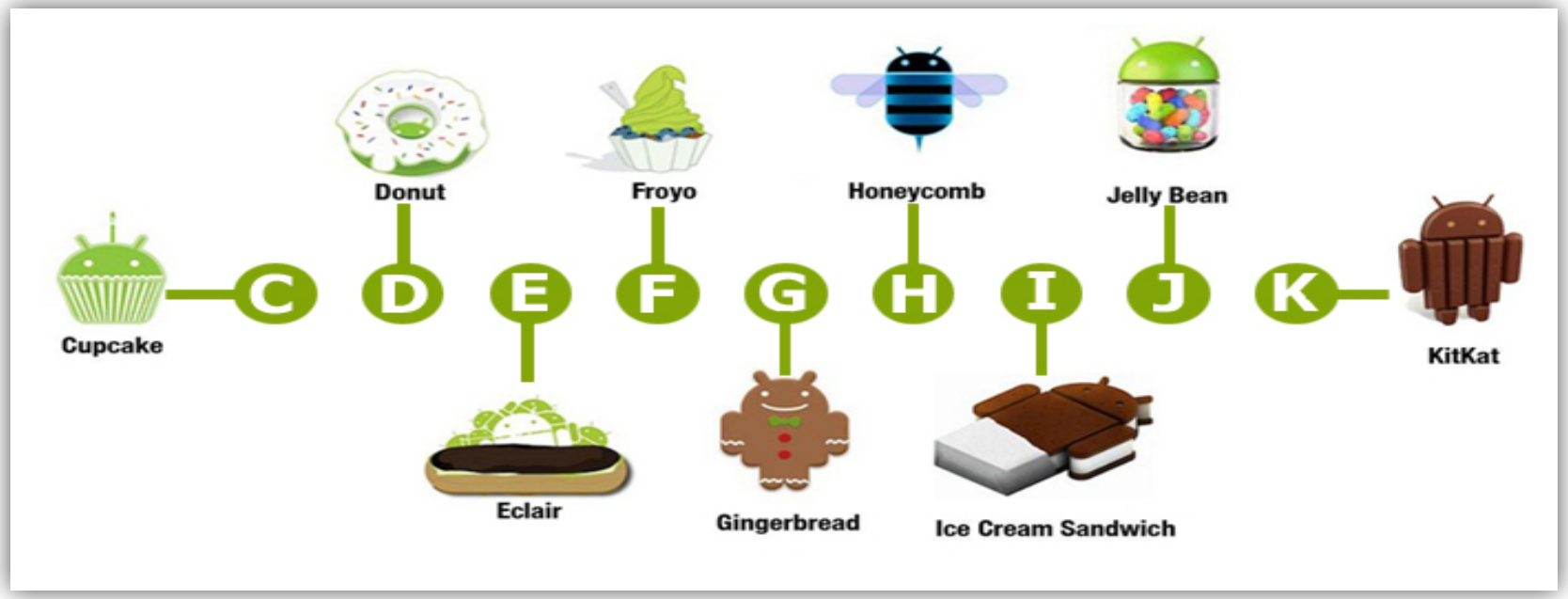
Android is:

- A free, open-source operating system for embedded devices
- An open-source development platform for creating applications
- Devices, particularly mobile phones, that run the Android operating system and the applications created for it

"The first truly open and comprehensive platform for mobile devices. It includes an operating system, user-interface and applications – all of the software to run a mobile phone but without the proprietary obstacles that have hindered mobile innovation."

– Andy Rubin, Where's My Gphone?

First bite on Android



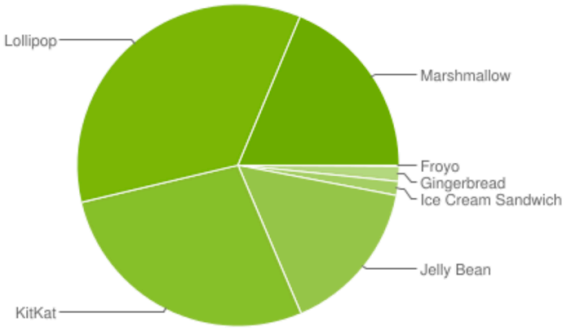
Lollipop → Marshmallow → Nougat

First bite on Android

Version	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	1.5%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.4%
4.1.x	Jelly Bean	16	5.6%
4.2.x		17	7.7%
4.3		18	2.3%
4.4	KitKat	19	27.7%
5.0	Lollipop	21	13.1%
5.1		22	21.9%
6.0	Marshmallow	23	18.7%

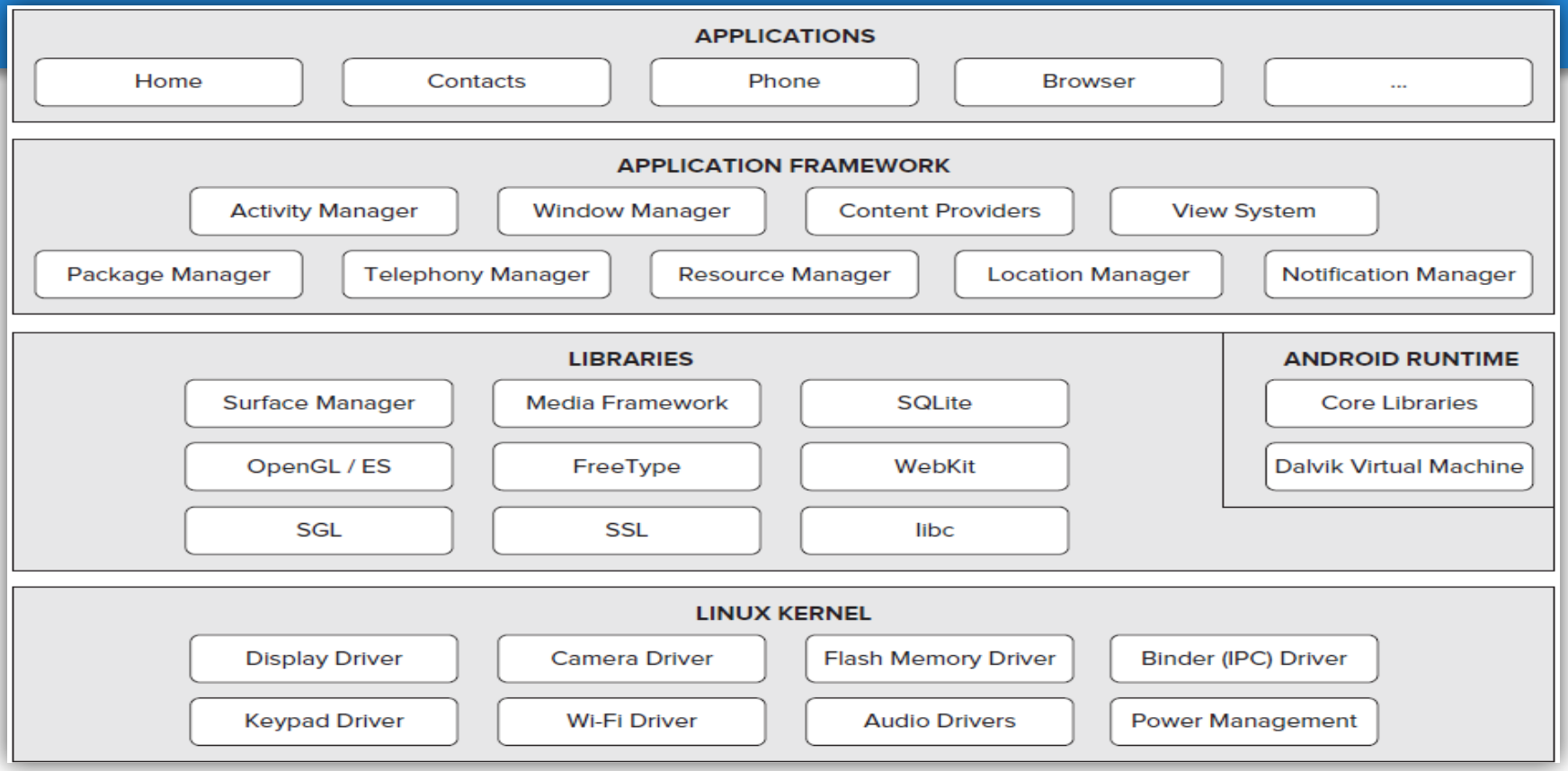
Data collected during a 7-day period ending on September 5, 2016.

Any versions with less than 0.1% distribution are not shown.



https://developer.android.com/about/dashboards/index.html

First bite on Android

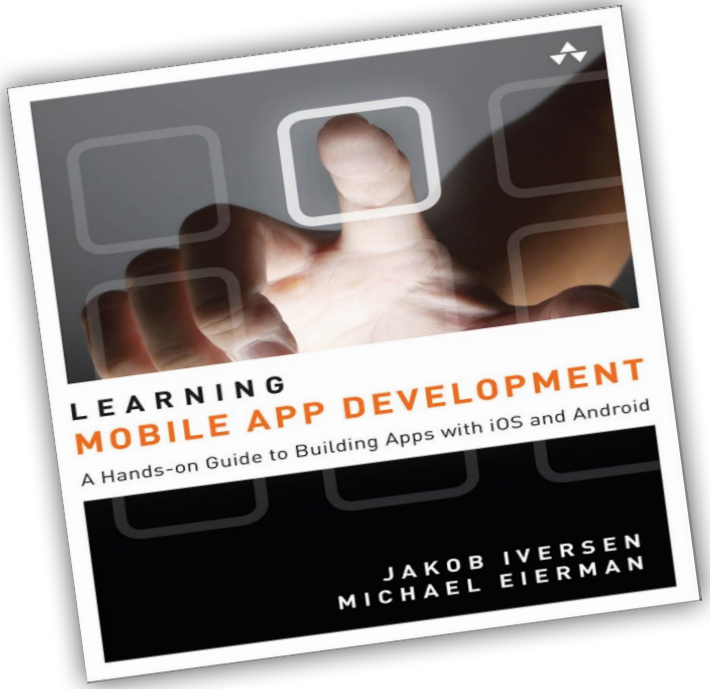


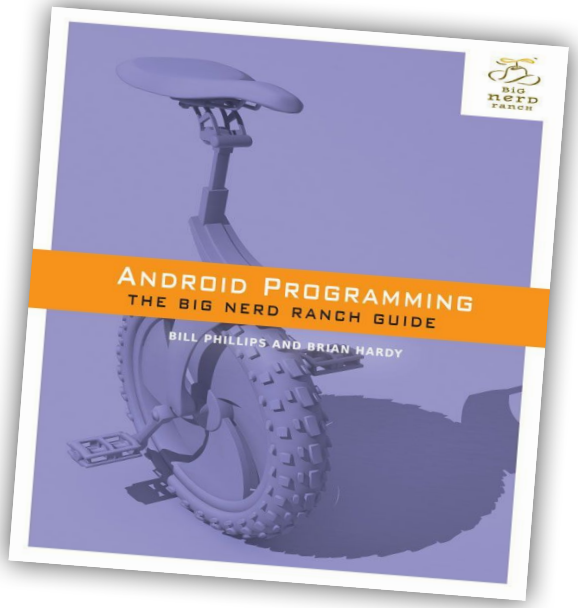
Tools and resources

Android Studio:

- Android SDK (tools, platform-tools/build-tools)
- A version of the Android platform
- A version of the Android system image for the emulator

Tools and resources





Tools and resources

Online resources:

- Stack Overflow
  - www.stackoverflow.com
- Google Android Training
  - http://developer.android.com/training/index.html
- Android Discuss
  - http://groups.google.com/group/android-discuss

Tools and resources

- Apps can be broadly classified into four distinct categories: native apps, generic mobile apps, dedicated web apps, and hybrid apps
- Native applications are usually developed using higher level programming languages, such as Java for Android, Objective-C for iOS

PhoneGap

Setting up IDE

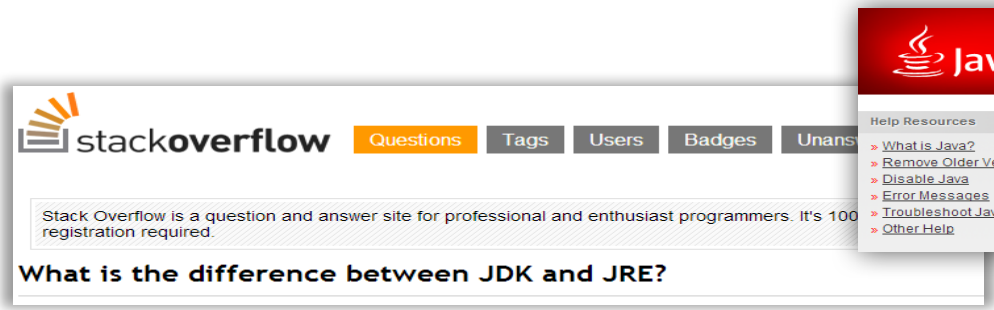
Step 1: install Java

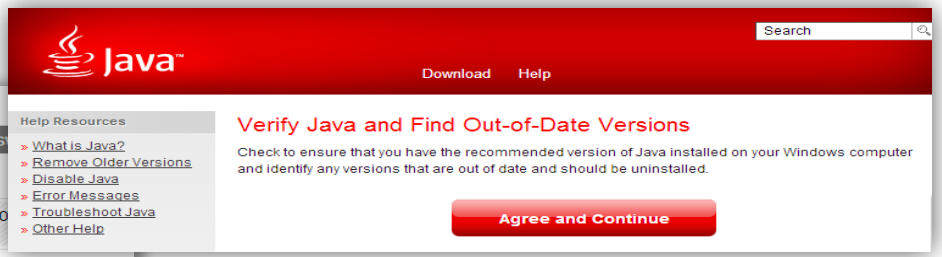
× JRE (Java Runtime Environment)

- Libraries
- Java Virtual Machine
- Components to run applets and java applications

✓ JDK (Java Development Toolkit)

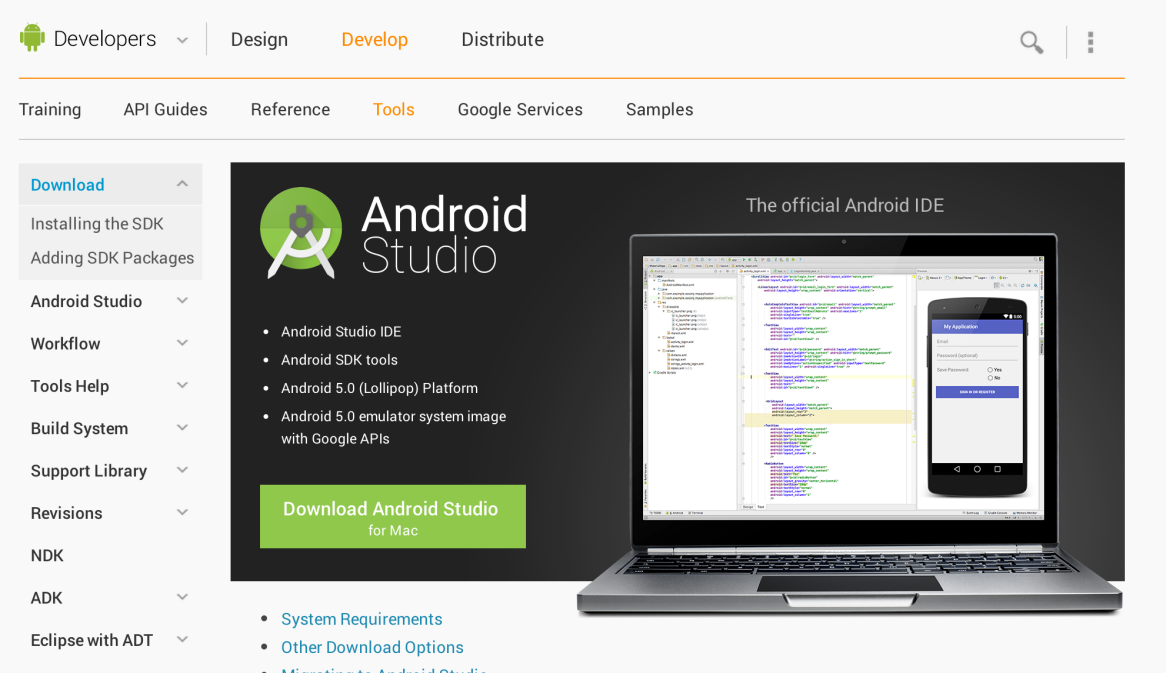
- Contains complete JRE
- Compilers
- Debuggers





Setting up IDE

Step 2: install Android Studio



Setting up IDE

Step 2: (alternative): IntelliJ

1. Download and install the Android SDK
2. Configure Android platforms
3. Download and install IntelliJ IDEA

https://confluence.jetbrains.com/display/IntelliJIDEA/Prerequisites+for+Android+development

IntelliJIDEA

Setting up IDE

Step 2 (alternative): the hard way

- Eclipse + ADT plugin
- Android SDK Tools
- Android Platform-tools
- A version of the Android platform
- A version of the Android system image for the emulator

Setting up IDE

Step 3: update SDK

- From Android Studio, select Tools > Android > SDK Manager.
- On Windows, double-click the SDK Manager.exe file at the root of the Android SDK directory.
- On Mac or Linux, open a terminal and navigate to the tools/ directory in the Android SDK, then execute android sdk.

http://developer.android.com/tools/help/sdk-manager.html

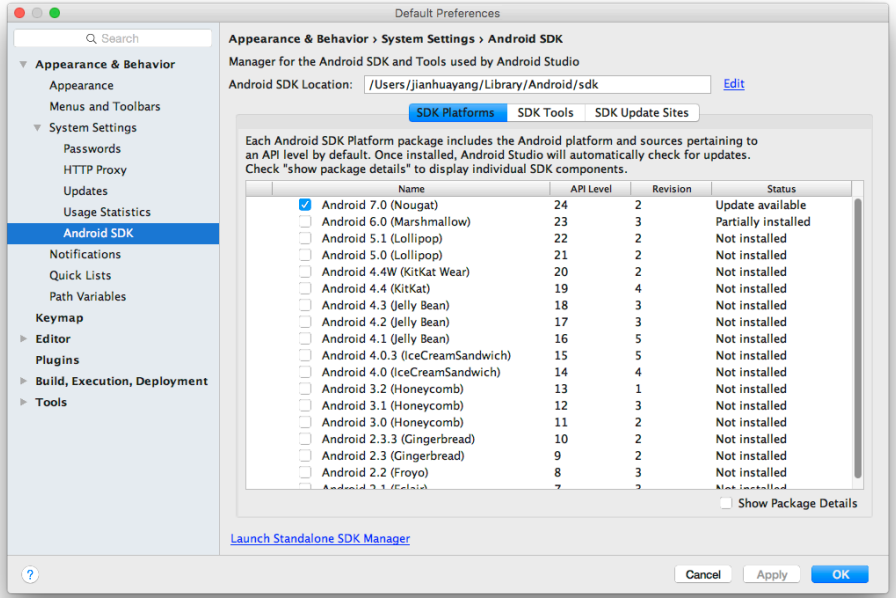
Setting up IDE

Step 3: update SDK

Bare minimum:

- The SDK Platform
- An emulator system image
- The Google APIs

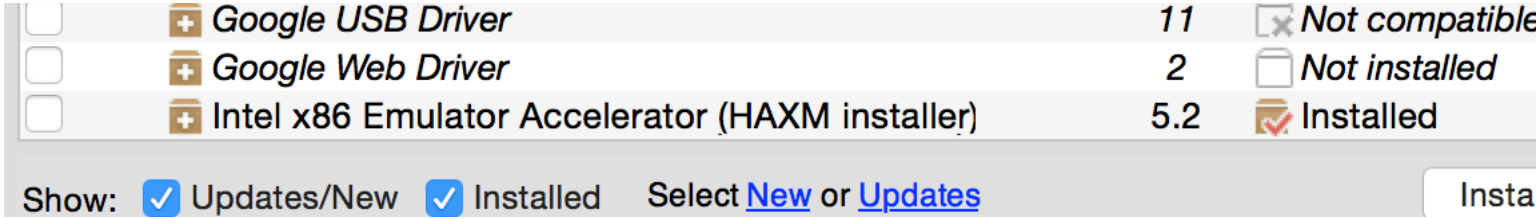
Full installation:



Setting up IDE

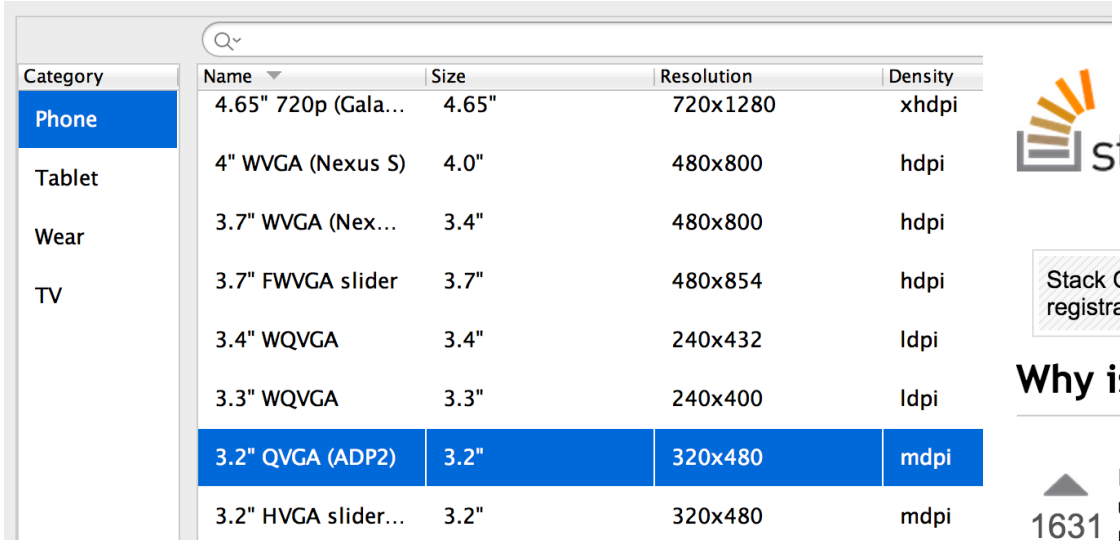
Step 3: update SDK

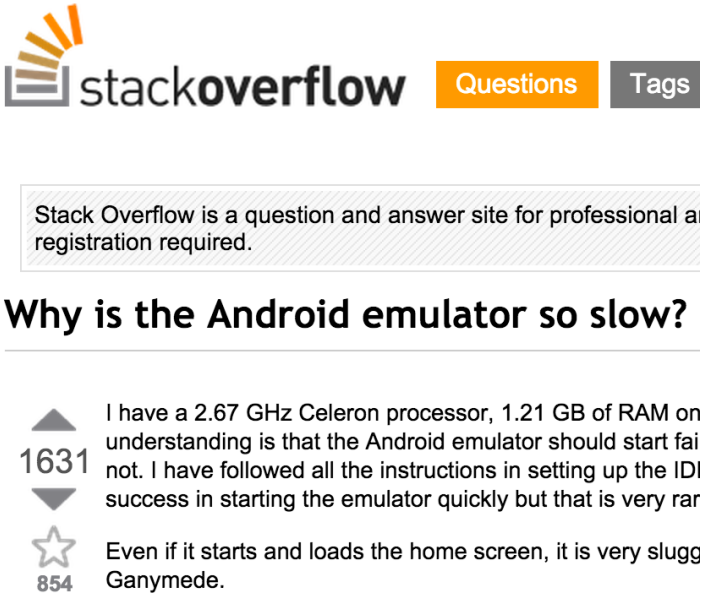
The Intel Emulator



Setting up IDE

Step 4: create Android virtual devices (AVDs)





Setting up IDE

Step 4: create Android virtual devices (AVDs)

- Start with one of the smaller devices because it's easier to scale up when developing the user interface than to scale down.
- AVDs can be quite slow, give it time.

Setting up IDE

Step 5 (optional): testing device

- Verify that your application is "debuggable" in your manifest or build.gradle file.
- Enable USB debugging on your device.
- Set up your system to detect your device.

```
android {
    buildTypes {
        debug {
            debuggable true
        }
```

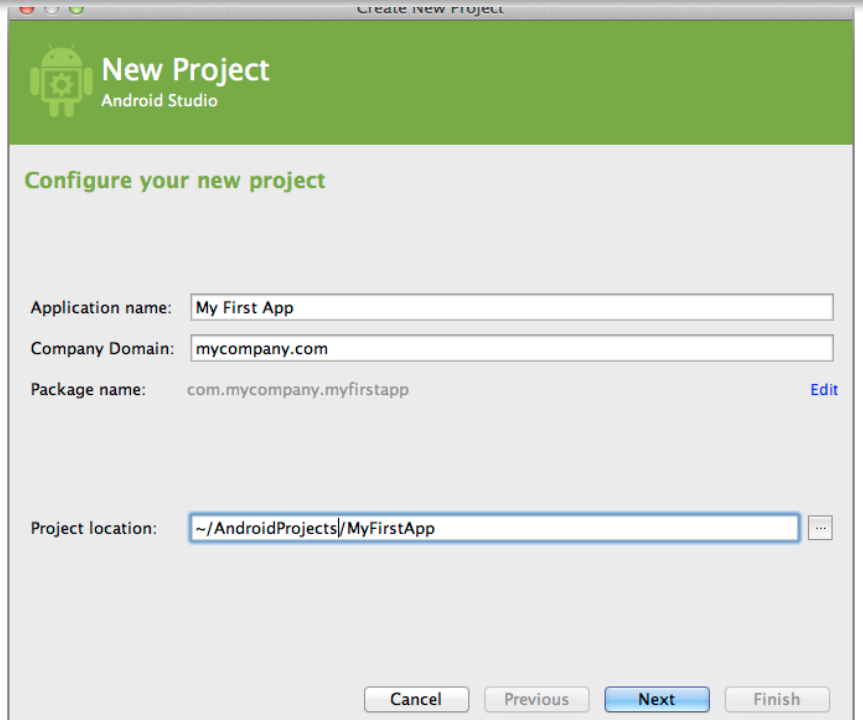
https://developer.android.com/tools/device.html

LAB 2

- Hello, Android!
- Android anatomy
- Coding the interface
- Coding the behaviour
- Create-modify-reuse

Hello, Android!

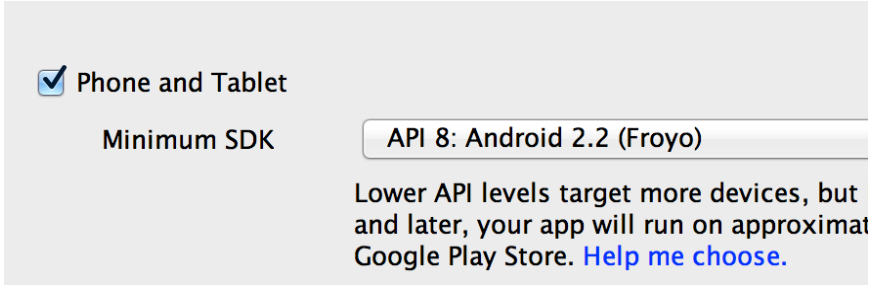
- **Application Name** is the app name that appears to users.
- **Company domain** provides a qualifier that will be appended to the package name; Android Studio will remember this qualifier for each new project you create.
- **Package Name** is the package namespace for your app (as packages in the Java).



http://developer.android.com/training/basics/firstapp/creating-project.html

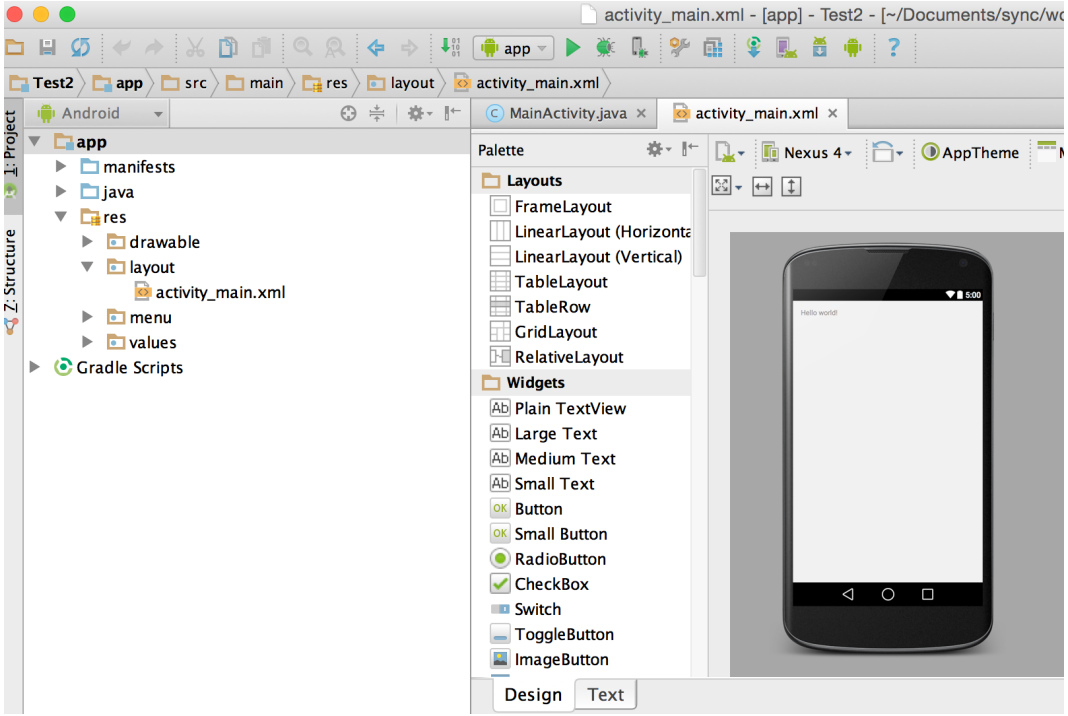
Hello, Android!

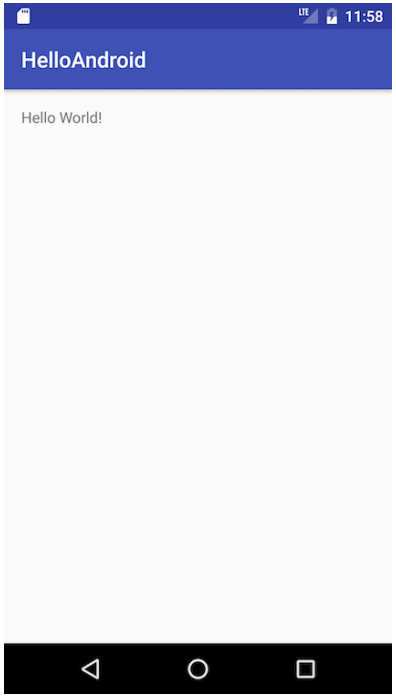
- **Minimum Required SDK** is the lowest version of Android that your app supports, indicated using the API level.
- **Target SDK** indicates the highest version of Android (also using the API level) with which you have tested with your application.
- **Compile With** is the platform version against which you will compile your app.



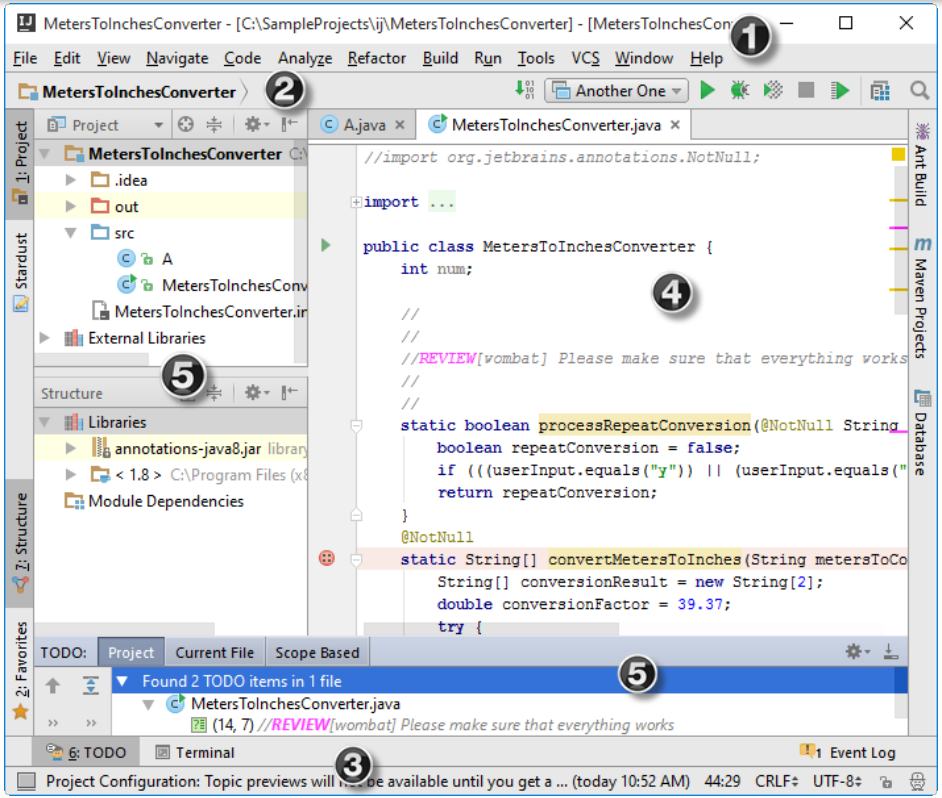
http://developer.android.com/training/basics/firstapp/creating-project.html

Hello, Android!

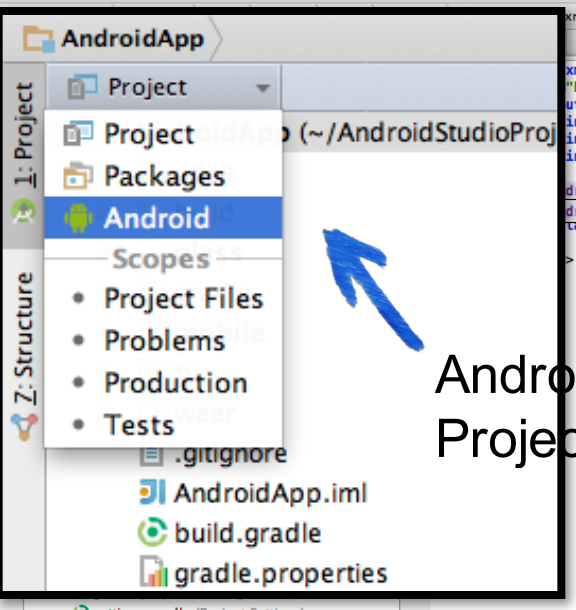




Android Studio

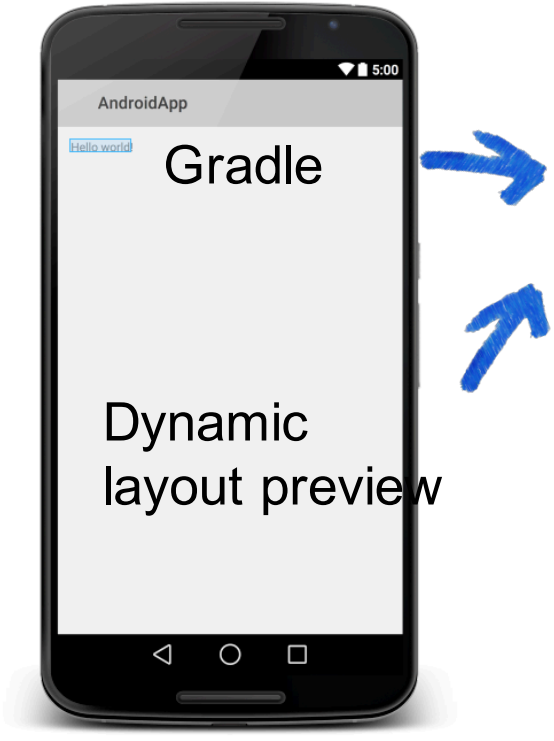


1. Menus and toolbars
2. Navigation bar
3. The status bar
4. The editor
5. Tool windows



Android
Project View

https://developer.android.com/tools/studio/index.html



Gradle

Dynamic
layout preview

Android anatomy

- manifest.xml

  This is the manifest file for your Android application. Here you specify the permissions needed by your application.

- java

  Contains the .java source files for your project.

- res

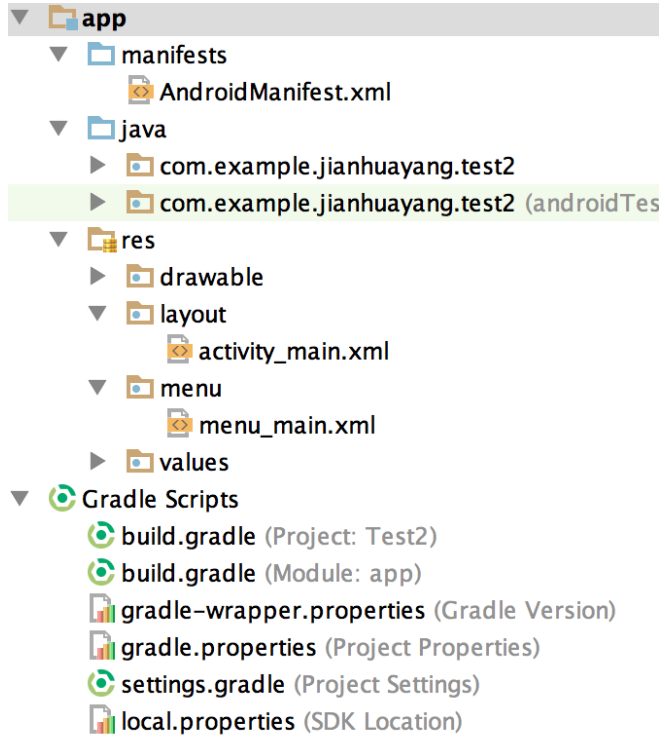
  This folder contains all the resources used in your application.

- Gradle

  Customizable properties for the build system.

- gen

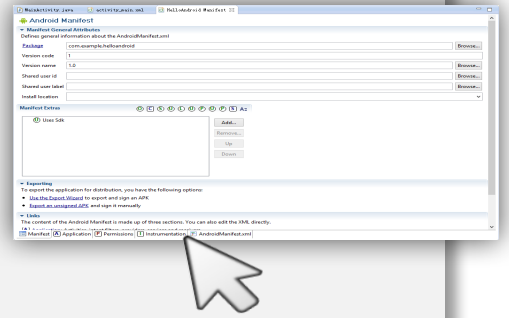
  Contains the R.java file, a compiler-generated file that references all the resources found in your project. You should not modify this file.



https://developer.android.com/tools/projects/index.html#ApplicationProjects

Android anatomy

```
1.  <?xml version="1.0" encoding="utf-8"?>
2.  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3.    package="com.example.helloandroid"
4.    android:versionCode="1"
5.    android:versionName="1.0" >
6.
7.    <uses-sdk
8.      android:minSdkVersion="8"
9.      android:targetSdkVersion="21" />
10.
11.   <application
12.     android:allowBackup="true"
13.     android:icon="@drawable/ic_launcher"
14.     android:label="@string/app_name"
15.     android:theme="@style/AppTheme" >
16.     <activity
17.       android:name=".MainActivity"
18.       android:label="@string/app_name" >
19.       <intent-filter>
20.         <action android:name="android.intent.action.MAIN" />
21.
22.         <category android:name="android.intent.category.LAUNCHER" />
23.       </intent-filter>
24.     </activity>
25.   </application>
26.
27. </manifest>
```



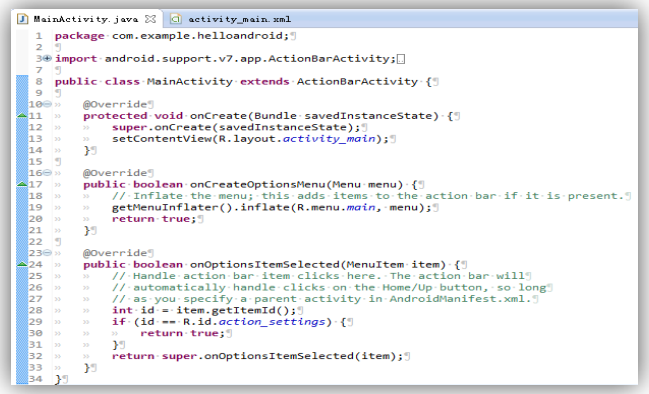
Manifest:

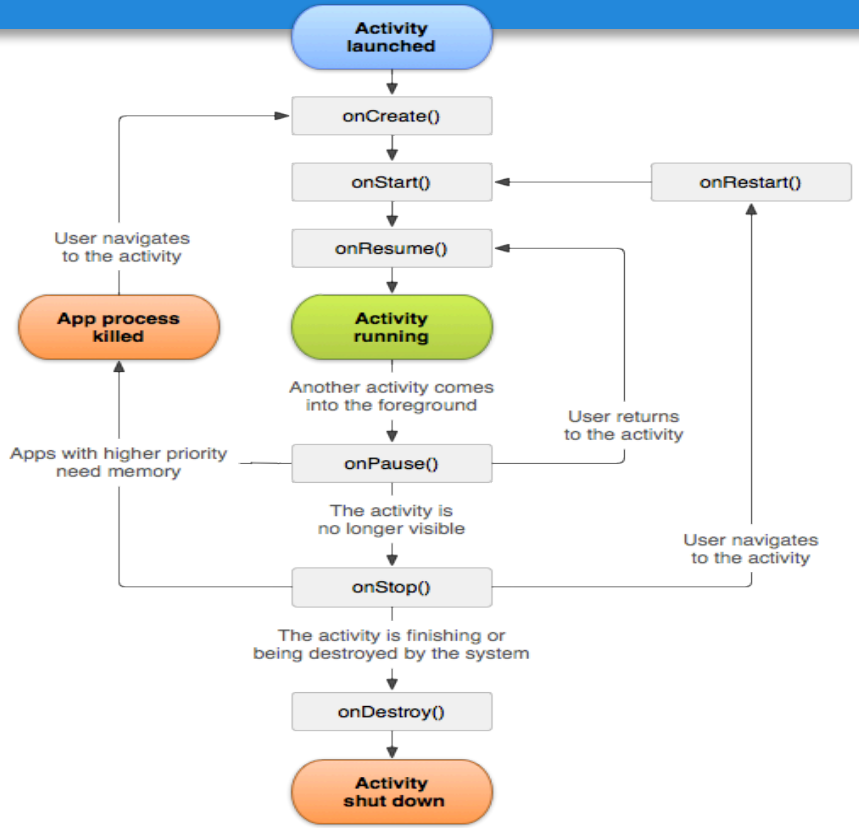
- App's components
- User permissions the app requires
- Minimum API Level required by the app
- Hardware and software features used or required by the app
- API libraries the app needs to be linked against

Android anatomy

Activity :

- An activity is an instance of Activity, a class in the Android SDK.
- An activity is responsible for managing user interaction with a screen of information.



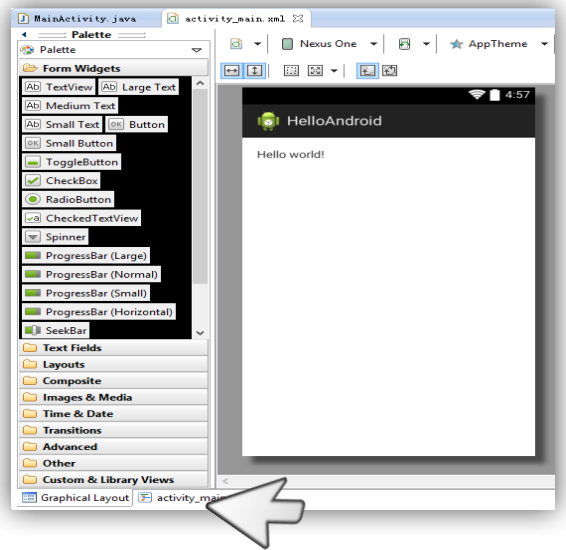


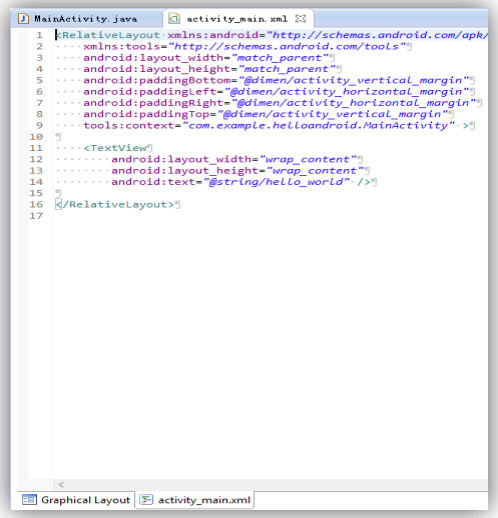
http://developer.android.com/reference/android/app/Activity.html
http://developer.android.com/guide/components/fundamentals.html

Android anatomy

Layout:

- A layout defines a set of user interface objects and their position on the screen.
- A layout is made up of definitions written in XML.

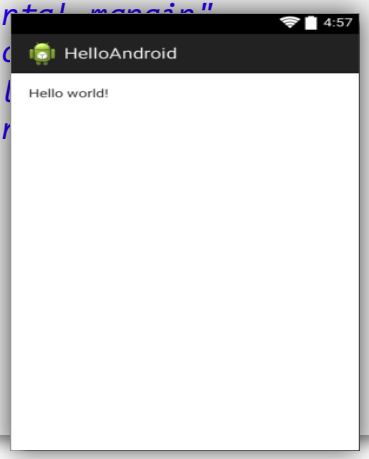


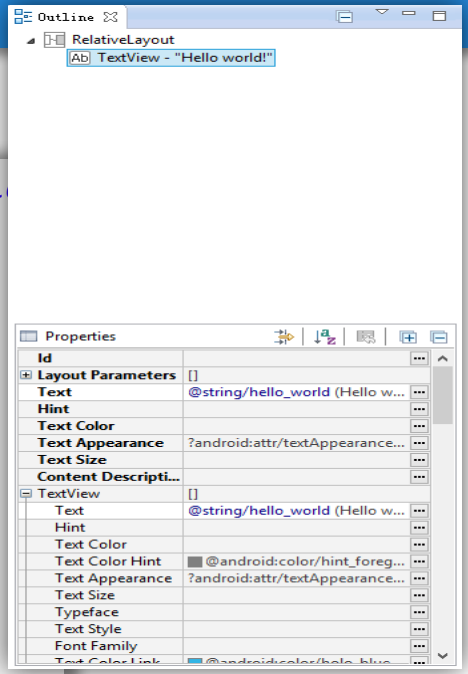


Coding the interface

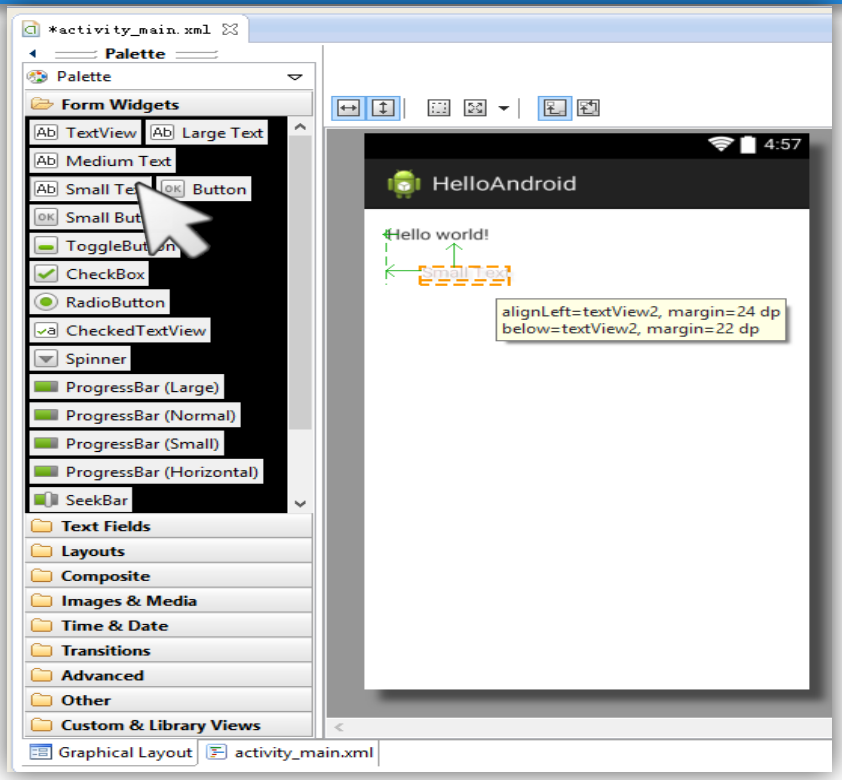
Auto-generated layout

```
1. <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/androi
2.     xmlns:tools="http://schemas.android.com/tools"
3.     android:layout_width="match_parent"
4.     android:layout_height="match_parent"
5.     android:paddingBottom="@dimen/activity_vertical_margin"
6.     android:paddingLeft="@dimen/activity_horizontal_margin"
7.     android:paddingRight="@dimen/activity_horizo
8.     android:paddingTop="@dimen/activity_vertical
9.     tools:context="com.example.helloandroid.Mai
10.
11.     <TextView
12.         android:layout_width="wrap_content"
13.         android:layout_height="wrap_content"
14.         android:text="@string/hello_world" />
15.
16.</RelativeLayout>
```

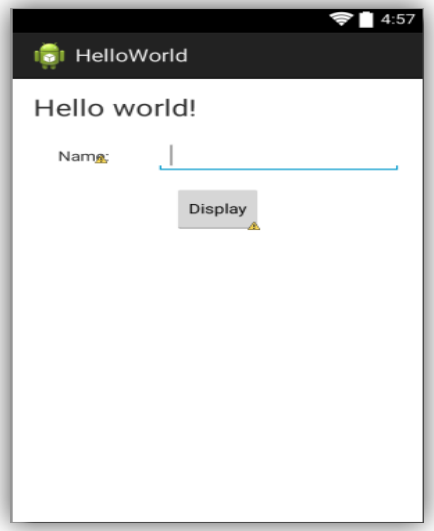




Coding the interface



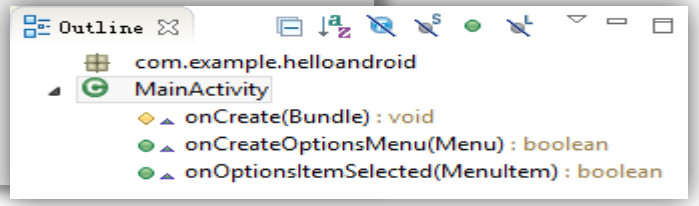
- SmallText
- PersonName
- Button



Coding the behaviour

Auto-generated activity

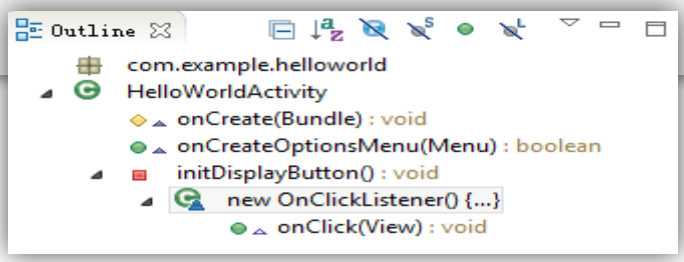
```
1. public class MainActivity extends ActionBarActivity {
2.
3.   @Override
4.   protected void onCreate(Bundle savedInstanceState) {
5.     super.onCreate(savedInstanceState);
6.     setContentView(R.layout.activity_main);
7.   }
8.
9.   @Override
10.  public boolean onCreateOptionsMenu(Menu menu) {
11.    getMenuInflater().inflate(R.menu.main, menu);
12.    return true;
13.  }
14.
15.  @Override
16.  public boolean onOptionsItemSelected(MenuItem item) {
17.    int id = item.getItemId();
18.    if (id == R.id.action_settings) {
19.      return true;
20.    }
21.    return super.onOptionsItemSelected(item);
22.  }
23.}
```



Coding the behaviour

setOnClickListener

```
1. private void initDisplayButton() {
2.   Button displayButton = (Button) findViewById(R.id.buttonDisplay);
3.   displayButton.setOnClickListener(new OnClickListener() {
4.
5.     @Override
6.     public void onClick(View arg0) {
7.       EditText editName = (EditText) findViewById(R.id.editTextName);
8.       TextView textDisplay = (TextView) findViewById(R.id.textViewDisplay);
9.       String nameToDisplay = editName.getText().toString();
10.      textDisplay.setText("Hello " + nameToDisplay);
11.
12.    }
13.
14.  });
15.}
```



Git – backgrounds

- Where is my code / I left it at home!
- What has changed and when?
- I've broken my program and can't fix it
- I need to work on a different computer
- My team members are all working on the same file!
- Who has modified the files and when?
- Who has been doing to the programming?

Revision control

- The management of changes to documents
- Each changes are usually identified by a revision number
- Each revision has a timestamp and the person responsible
- Revisions can be compared, restored, and merged

What is Git?

- Git is a version control system (VCS)
- It can help you keep track of files that are frequently changed
- It supports distributed development
- Its Open Source (you can download and install it for free)
- Alternatives: Mercurial, subverson (SVN)

Git terms

- HEAD: the current commit your repo is on.
- Master: the name of the default branch that Git creates for you when first creating a repo.
- Origin: the default name that Git gives to your main remote repo.

http://stackoverflow.com/questions/8196544/what-are-the-git-concepts-of-head-master-origin

Git Hosting Services

- Many online hosting services
- Some are free
- Some cost money
- Popular hosting services:
  - GitHub
  - BitBucket
  - GitLab

A typical workflow

- Create a project on C9 using the Git option.
- Add new files and add them to the local Git repository.
- Commit changes as you work.
- Create a private repository on GitHub.
- Copy the repository address (HTTPS).
- Push changes to the remote repository at the ned of your session.

A typical workflow

- Create a local git repository
  - git init
- Create an empty remote (GitHub)
- Add this as a new remote
  - git remote add origin git@../repo.git
- Push commits
  - git push origin master

Create-modify-reuse

- Code examples from “Learning Mobile App Development”
- Downloadable at https://github.com/LearningMobile/BookApps
- Import, look for ‘gradle’ file