Dev Environment & App Workflow

ICT2105 Mobile Application Development

Jeannie S. Lee

Spring 2020

Overview

Development Environment

Emulation vs Simulation

Android Monitor & Other Tools

App Workflow

Version Control

Continuous Integration

Development Environment

Workbench for writing Android applications

Oracle Java JDK 8

Android Studio IDE & SDK Tools

Android Emulator

Debugging

Android Device Monitor

Android Profiler

Other Development Tools

Android Emulator

Virtual Android Device - Sometimes slow (why?)

Some features are unavailable

No Bluetooth or USB

Advanced emulation features

Different network speeds

Battery power

Location coordinates

Incoming sms or phone calls

https://developer.android.com/studio/run/emulator.html

Emulator vs. Simulator

Emulation: system that behaves **exactly like** something else, and abides by all the rules

Replicates as on the original, used as a substitute

Simulation: system that behaves **similar** to something else, but possibly implemented in an entirely different way

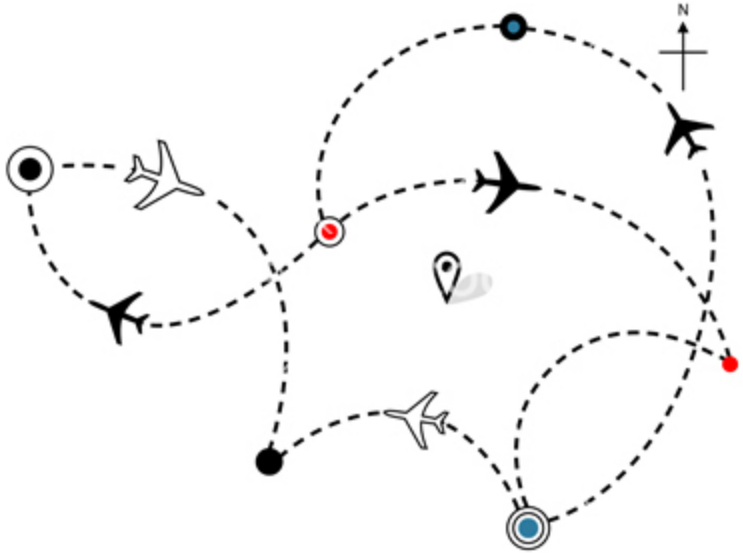
Provides basic behavior of the system but may not abide by all the rules; gives a basic idea

Is a model for analysis and study

Flight Simulator Example

Flight simulator: Looks and feels like flying an airplane, but you are disconnected from the reality of flying the plane

If the flight simulator brought you from Point A to Point B, that's an **emulation**

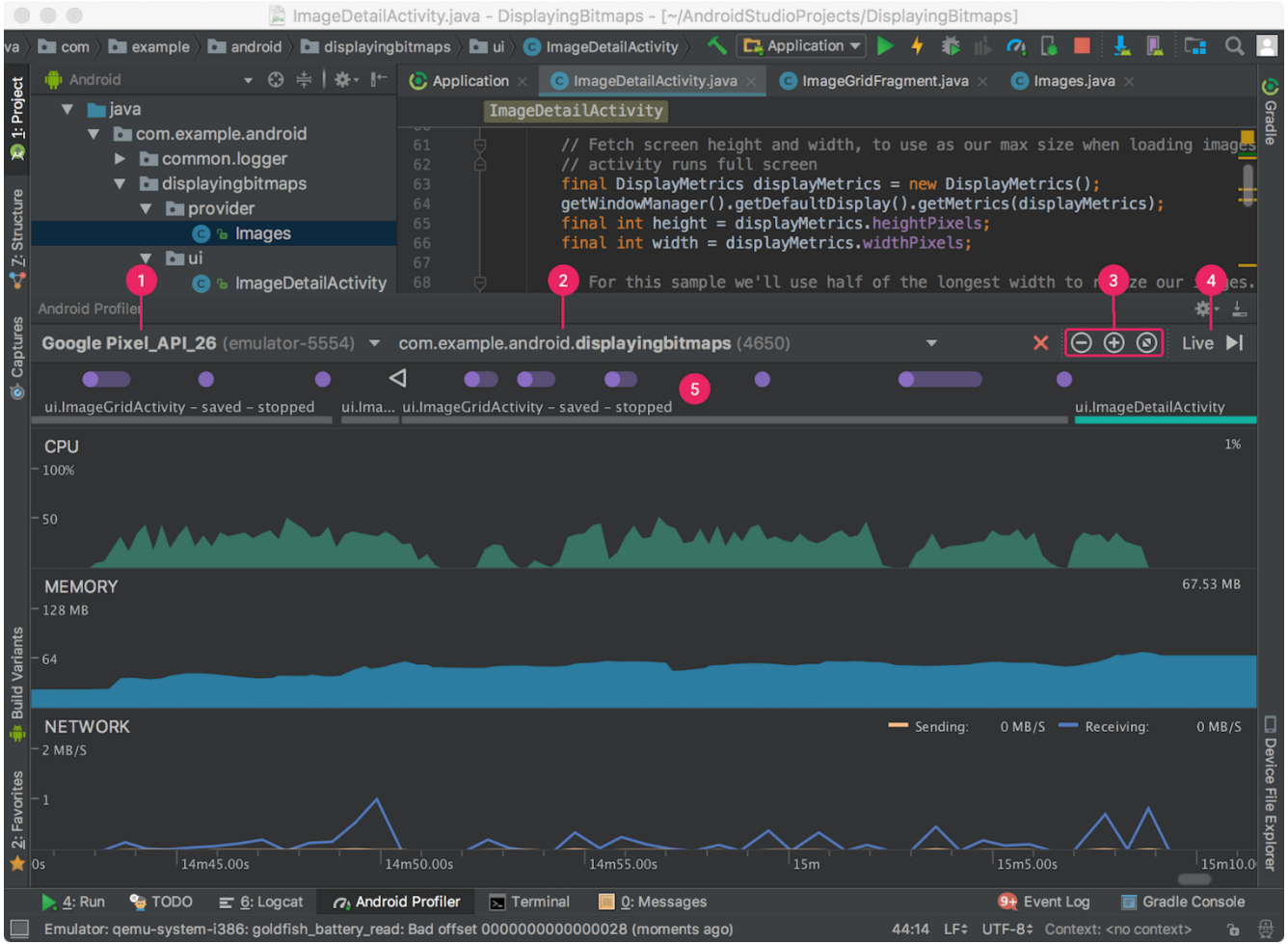


Android Profiler

Tools for monitoring and analyzing application behavior, CPU and memory usage, processes and threads, etc.

https://developer.android.com/studio/profile/android-profiler

https://developer.android.com/studio/profile/am-basics


ImageDetailActivity.java - DisplayingBitmaps - [~/AndroidStudioProjects/DisplayingBitmaps]
com
example
android
displayingbitmaps
ui
ImageDetailActivity
Application
Android
java
com.example.android
common.logger
displayingbitmaps
provider
Images
ui
ImageDetailActivity
ImageGridFragment.java
Images.java
// Fetch screen height and width, to use as our max size when loading images
// activity runs full screen
final DisplayMetrics displayMetrics = new DisplayMetrics();
getWindowManager().getDefaultDisplay().getMetrics(displayMetrics);
final int height = displayMetrics.heightPixels;
final int width = displayMetrics.widthPixels;
Android Profiler
Google Pixel_API_26 (emulator-5554)
com.example.android.displayingbitmaps (4650)
Live
1
2
3
4
5
ui.ImageGridActivity – saved – stopped
ui.Ima...
ui.ImageGridActivity – saved – stopped
ui.ImageDetailActivity
CPU
100%
50
1%
MEMORY
128 MB
64
67.53 MB
NETWORK
2 MB/S
1
Sending: 0 MB/S
Receiving: 0 MB/S
0s
14m45.00s
14m50.00s
14m55.00s
15m
15m5.00s
15m10.0
4: Run
TODO
6: Logcat
Android Profiler
Terminal
0: Messages
Event Log
Gradle Console
Emulator: qemu-system-i386: goldfish_battery_read: Bad offset 0000000000000028 (moments ago)
44:14
LF
UTF-8
Context: <no context>
1: Project
7: Structure
Captures
Build Variants
2: Favorites
Gradle
Device File Explorer

App Workflow

1. Define Resources & XML
2. Implement application classes in Code (Java)
3. Package the application

    Converted to bytecode

4. Install & run application

    VM executes bytecode

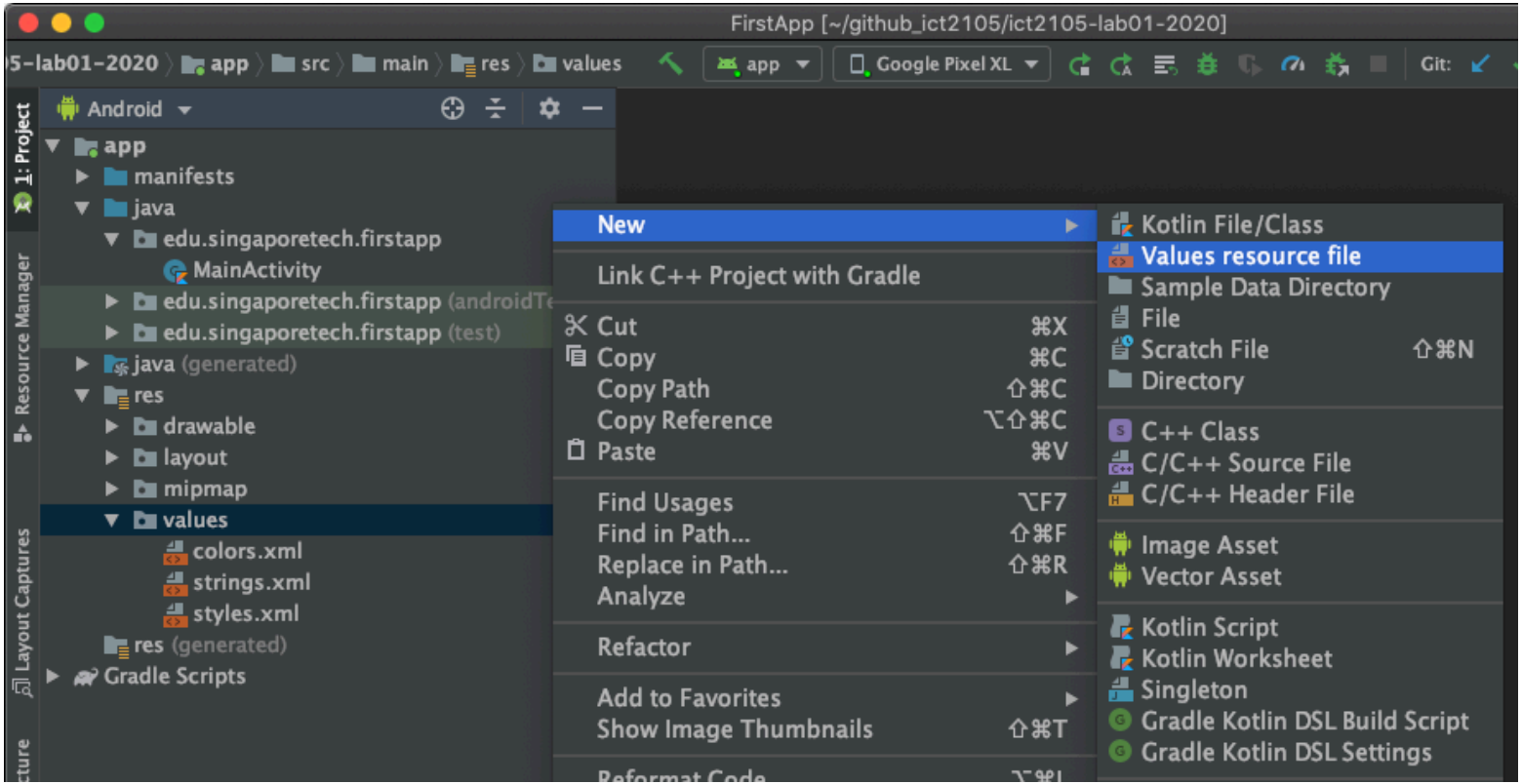
1. Define Resources & XML

Non-source code items

Many different resource types

- Layouts (XML)
- Strings
- Images
- Menus
- Animations

Allows customization for different devices, regions, languages, etc.


FirstApp [~/github_ict2105/ict2105-lab01-2020]
5-lab01-2020 app src main res values
app
Google Pixel XL
Git:
Android
1: Project
Resource Manager
Layout Captures
app
manifests
java
edu.singaporetech.firstapp
MainActivity
edu.singaporetech.firstapp (androidTe
edu.singaporetech.firstapp (test)
java (generated)
res
drawable
layout
mipmap
values
colors.xml
strings.xml
styles.xml
res (generated)
Gradle Scripts
New
Link C++ Project with Gradle
Cut ⌘X
Copy ⌘C
Copy Path ⇧⌘C
Copy Reference ⌥⇧⌘C
Paste ⌘V
Find Usages ⌥F7
Find in Path... ⇧⌘F
Replace in Path... ⇧⌘R
Analyze
Refactor
Add to Favorites
Show Image Thumbnails ⇧⌘T
Kotlin File/Class
Values resource file
Sample Data Directory
File
Scratch File ⇧⌘N
Directory
C++ Class
C/C++ Source File
C/C++ Header File
Image Asset
Vector Asset
Kotlin Script
Kotlin Worksheet
Singleton
Gradle Kotlin DSL Build Script
Gradle Kotlin DSL Settings

R.java

Resources are compiled into the R.java class

Auto-Generated: DO NOT EDIT!

Java code uses the R class to access resources

Use `findViewById()` and Resources object to get access to the resources:

E.g. `Button = (Button)findViewById(R.id.button);`

E.g. `getResources().getString(R.string.hello);`

Strings

String, String Array, Plurals

Stored in res/values/*.xml

Specified in XML

```
<string name="hello">Hello World!</string>
```

Other resources access through

```
@string/string_name
```

Java/Kotlin access through

```
R.string.string_name
```

UI Layout in XML

Specify UI elements and layout in XML

Separation of code from views

Can use graphical layout tools

Stored in `res/layout/*.xml`

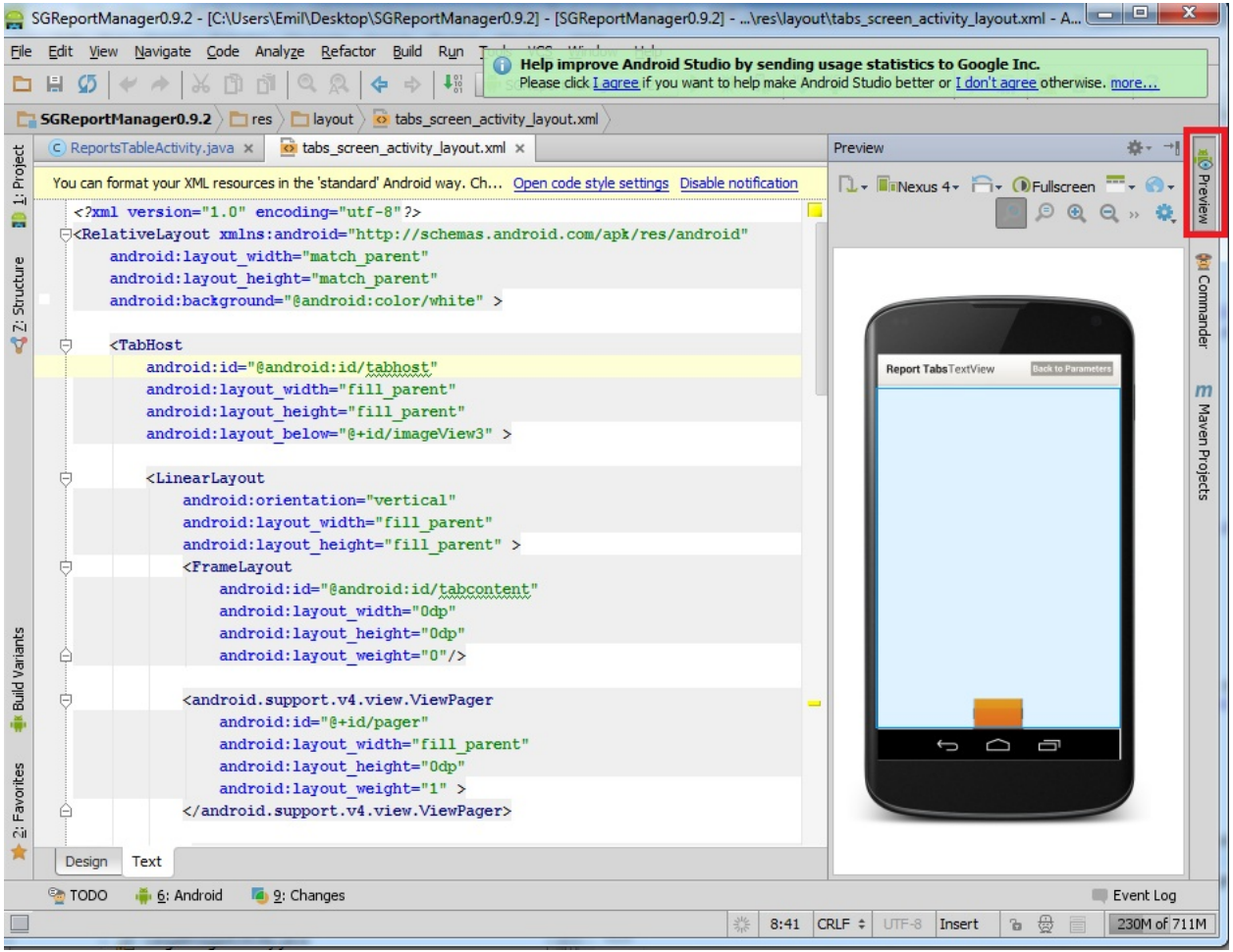
Other resources access through

```
@layout/layout_name
```

Java/Kotlin access through

```
R.layout.layout_name
```

Different layout files for different devices, orientation, screen size


SGReportManager0.9.2 - [C:\Users\Emil\Desktop\SGReportManager0.9.2] - [SGReportManager0.9.2] - ...\res\layout\tabs_screen_activity_layout.xml - A...
File Edit View Navigate Code Analyze Refactor Build Run
Help improve Android Studio by sending usage statistics to Google Inc.
Please click I agree if you want to help make Android Studio better or I don't agree otherwise. more...
SGReportManager0.9.2 res layout tabs_screen_activity_layout.xml
ReportsTableActivity.java
tabs_screen_activity_layout.xml
You can format your XML resources in the 'standard' Android way. Ch... Open code style settings Disable notification
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:background="@android:color/white" >
<TabHost
android:id="@android:id/tabhost"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:layout_below="@+id/imageView3" >
<LinearLayout
android:orientation="vertical"
android:layout_width="fill_parent"
android:layout_height="fill_parent" >
<FrameLayout
android:id="@android:id/tabcontent"
android:layout_width="0dp"
android:layout_height="0dp"
android:layout_weight="0" />
<android.support.v4.view.ViewPager
android:id="@+id/pager"
android:layout_width="fill_parent"
android:layout_height="0dp"
android:layout_weight="1" >
</android.support.v4.view.ViewPager>
Design
Text
Preview
Nexus 4
Fullscreen
Report Tabs TextView
Back to Parameters
1: Project
7: Structure
Build Variants
2: Favorites
Preview
Commander
Maven Projects
TODO
6: Android
9: Changes
Event Log
8:41
CRLF
UTF-8
Insert
230M of 711M

2. Implement Code

Implement Java/Kotlin classes

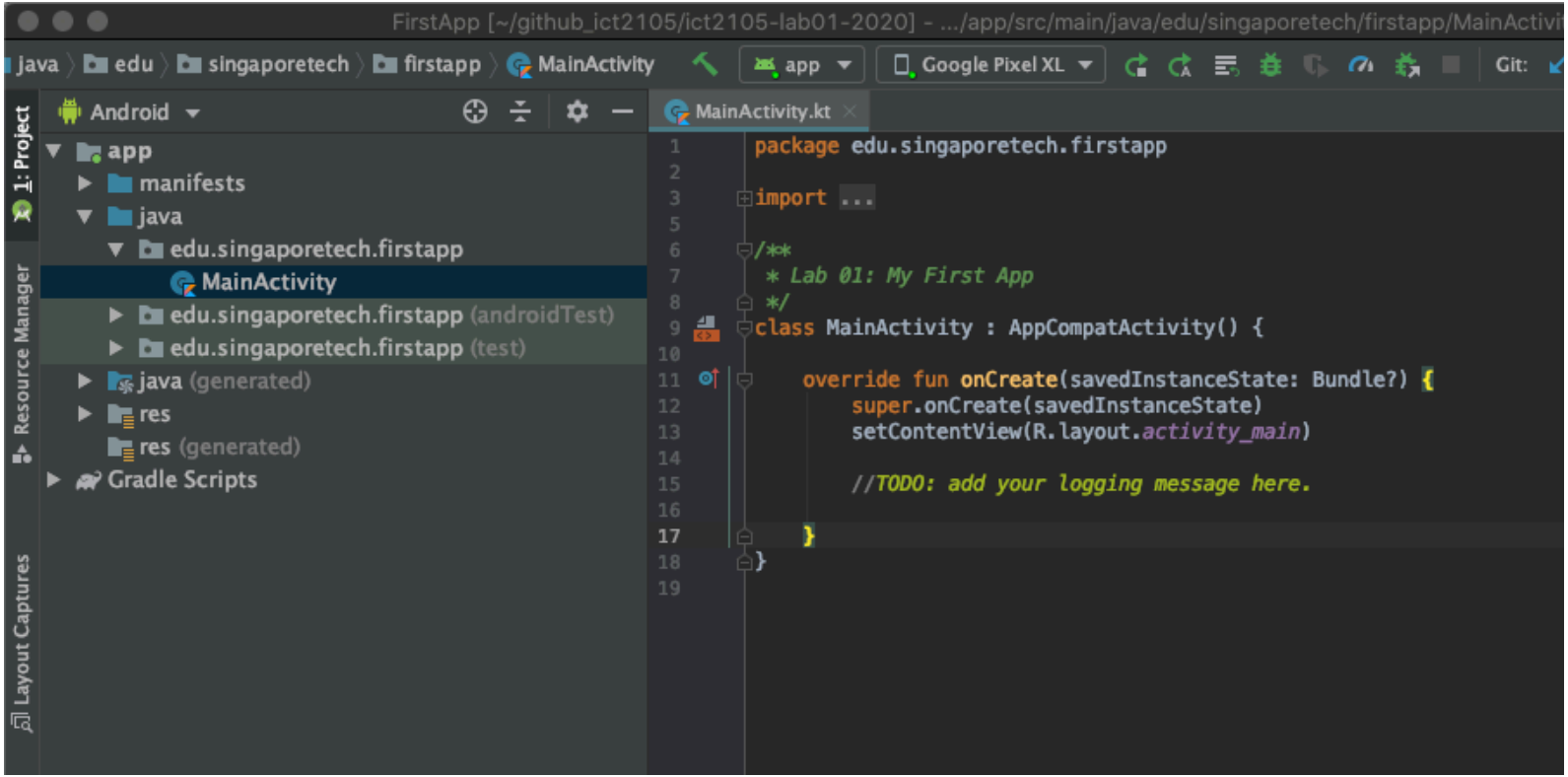
Usually requires at least one Activity

Activity initialization in `onCreate()`

2. Implement Code

Within `onCreate()`

- Restore saved state
- Set content view
- Initialize UI Elements
- Link UI Elements to code


FirstApp [~/github_ict2105/ict2105-lab01-2020] - .../app/src/main/java/edu/singaporetech/firstapp/MainActivi
java
edu
singaporetech
firstapp
MainActivity
app
Google Pixel XL
Git:
Android
app
manifests
java
edu.singaporetech.firstapp
MainActivity
edu.singaporetech.firstapp (androidTest)
edu.singaporetech.firstapp (test)
java (generated)
res
res (generated)
Gradle Scripts
MainActivity.kt
1: Project
Resource Manager
Layout Captures
package edu.singaporetech.firstapp
import ...
/**
* Lab 01: My First App
*/
class MainActivity : AppCompatActivity() {
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
setContentView(R.layout.activity_main)
//TODO: add your logging message here.
}
}

3. Package Application

Required app information is in

`AndroidManifest.xml`

Java/Kotlin classes, resources, etc. packaged into .APK file

AndroidManifest.xml

Contains essential information the system must have before running the application:

Application Name

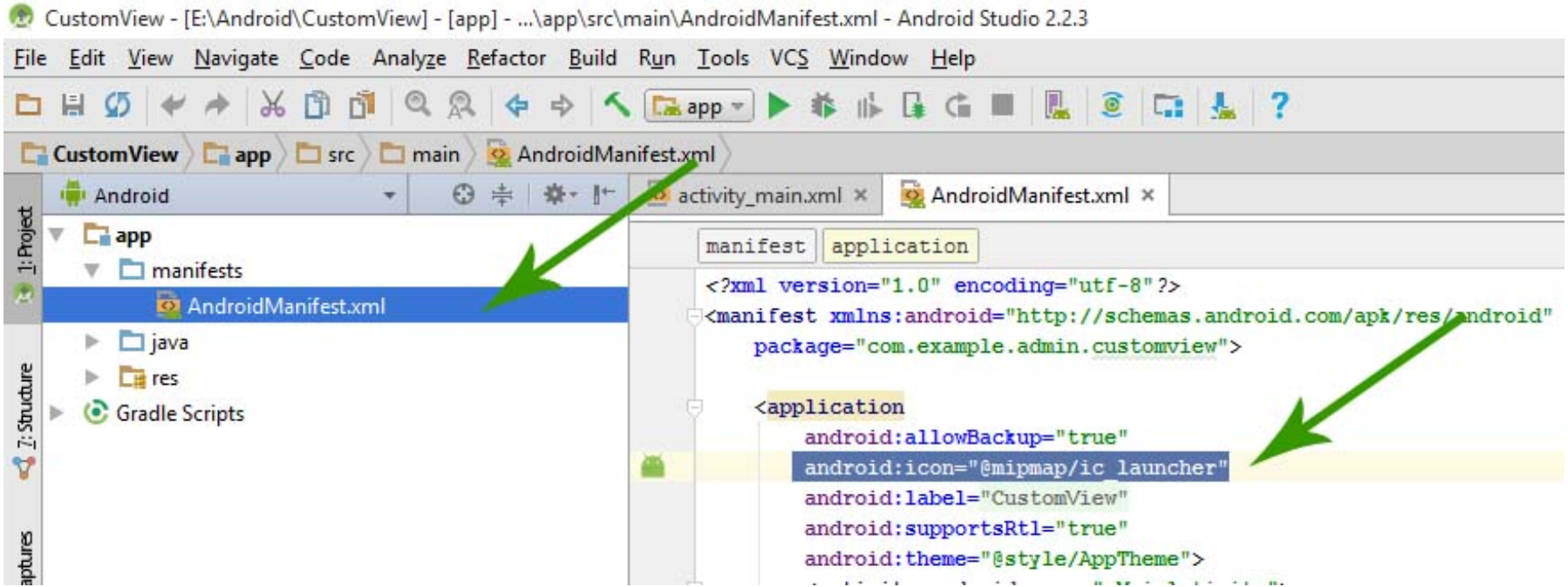
Components (Activities, Services, etc)

Permissions

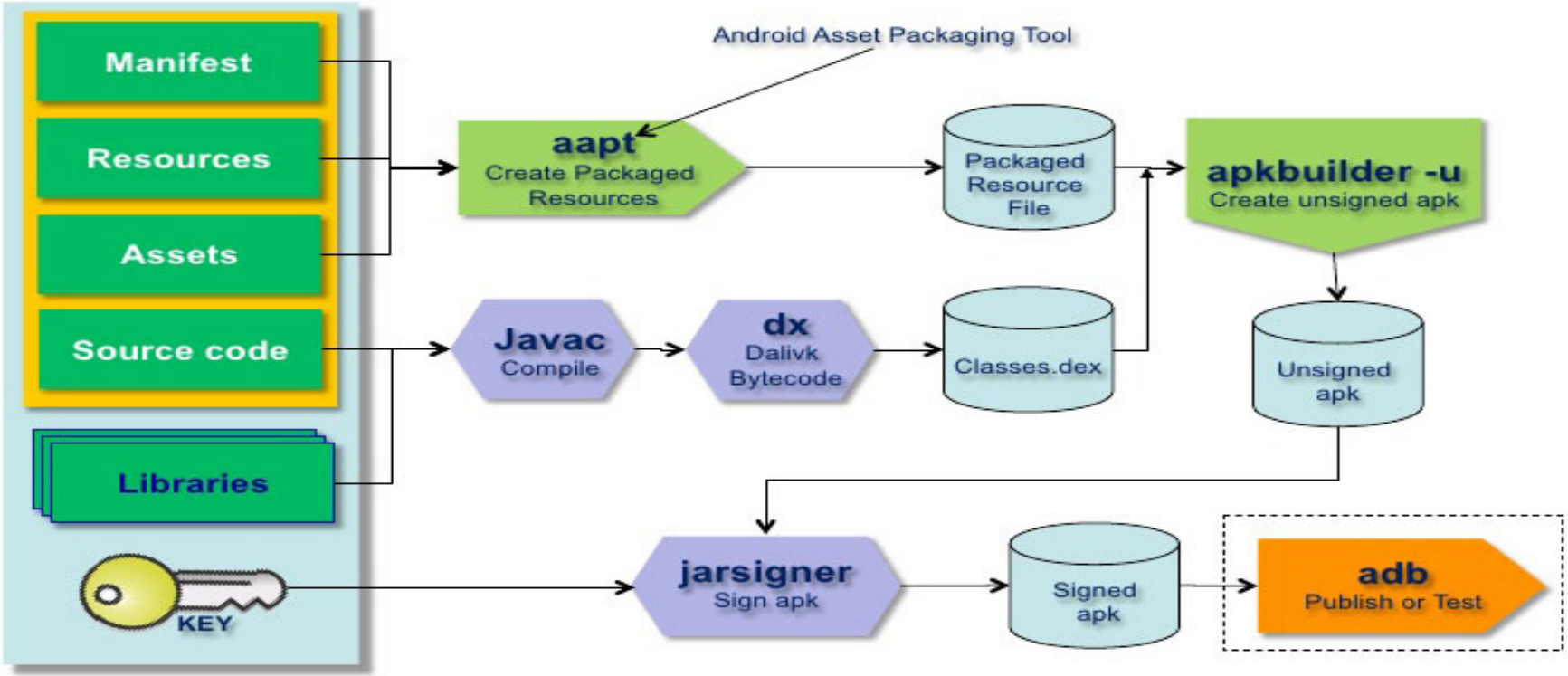
Other Features

Minimum API Level

AndroidManifest.xml



App Packaging Workflow



4. Install & Run

Hit “Play” button in Android Studio, run in the emulator

Enable USB Debugging on real device

Command Line:

```
adb install <filename.apk>
```

Version Control

System that records changes to a file or set of files over time so that you can recall specific **versions** later

Usually applies to code

Can do this with any file on the computer

Centralized Version Control

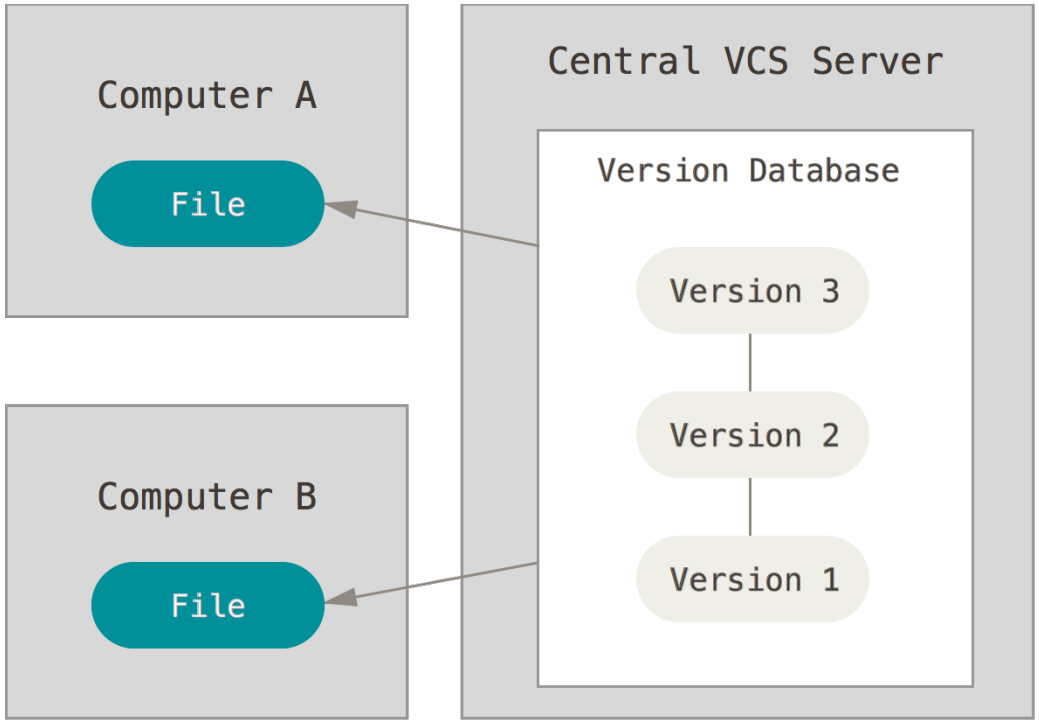
Single server that contains all the versioned files

A number of clients that check out files from that central place

Possibly single point of failure

E.g. CVS, Subversion, Perforce

Centralized Version Control



Distributed Version Control

Clients fully mirror the repository

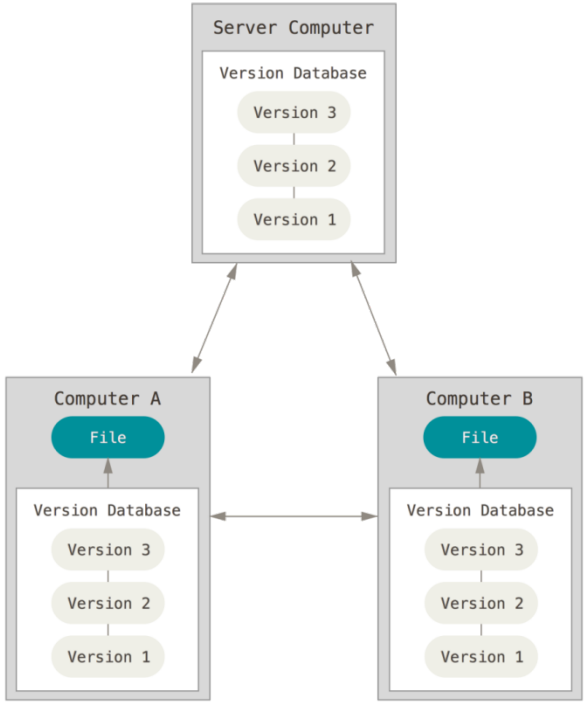
If any server dies, any of the client repositories can be copied back up to the server to restore it

Every clone is really a full backup of all the data

E.g. Git, Mercurial, Bazaar or Darcs

We’re using Git!

Distributed Version Control



Coding Conventions

Java https://google.github.io/styleguide/javaguide.html

Kotlin https://developer.android.com/kotlin/style-guide

Improve the readability of software

Highlight potential bugs or avoid mistakes

Reduce training and learning curve

Java Naming Conventions

- Match a class name with its file name [High]

WRONG

```
/**
 * Copyright Notice
 * Filename: Hello.java
 */
package learning.com.myprograms;
public class HelloWorld {

}
```

RIGHT

```
/**
 * Copyright Notice
 * Filename: HelloWorld.java
 */
package learning.com.myprograms;
public class HelloWorld {

}
```

Java Naming Conventions *contd…*

- Group operations with the same name together [Low]

WRONG

```
package learning.com.myprograms;
public class HelloWorld {
    void operation() {
    }
    void function() {
    }
    void operation(int param) {
    }
}
```

RIGHT

```
package learning.com.myprograms;
public class HelloWorld {
    void operation() {
    }
    void operation(int param) {
    }
    void function() {
    }
}
```

Kotlin Naming Conventions

Naming

If a source file contains only a single top-level class, the file name should reflect the case-sensitive name plus the `.kt` extension. Otherwise, if a source file contains multiple top-level declarations, choose a name that describes the contents of the file, apply PascalCase, and append the `.kt` extension.

```
// MyClass.kt
class MyClass { }
```

```
// Bar.kt
class Bar { }
fun Runnable.toBar(): Bar = // …
```

```
// Map.kt
fun <T, O> Set<T>.map(func: (T) -> O): List<O> = // …
fun <T, O> List<T>.map(func: (T) -> O): List<O> = // …
```

Kotlin Coding Conventions

Braces

Braces are not required for `when` branches and `if` statement bodies which have no `else if/else` branches and which fit on a single line.

```
if (string.isEmpty()) return

when (value) {
    0 -> return
    // …
}
```

Braces are otherwise required for any `if`, `for`, `when` branch, `do`, and `while` statements, even when the body is empty or contains only a single statement.

```
if (string.isEmpty())
    return  // WRONG!

if (string.isEmpty()) {
    return  // Okay
}
```

Continuous Integration

Development practice that requires developers to **integrate** code into a shared repository several times a day

Each check-in is then verified by an automated build, allowing teams to detect problems early

Build: Compile all the code, libraries, resources

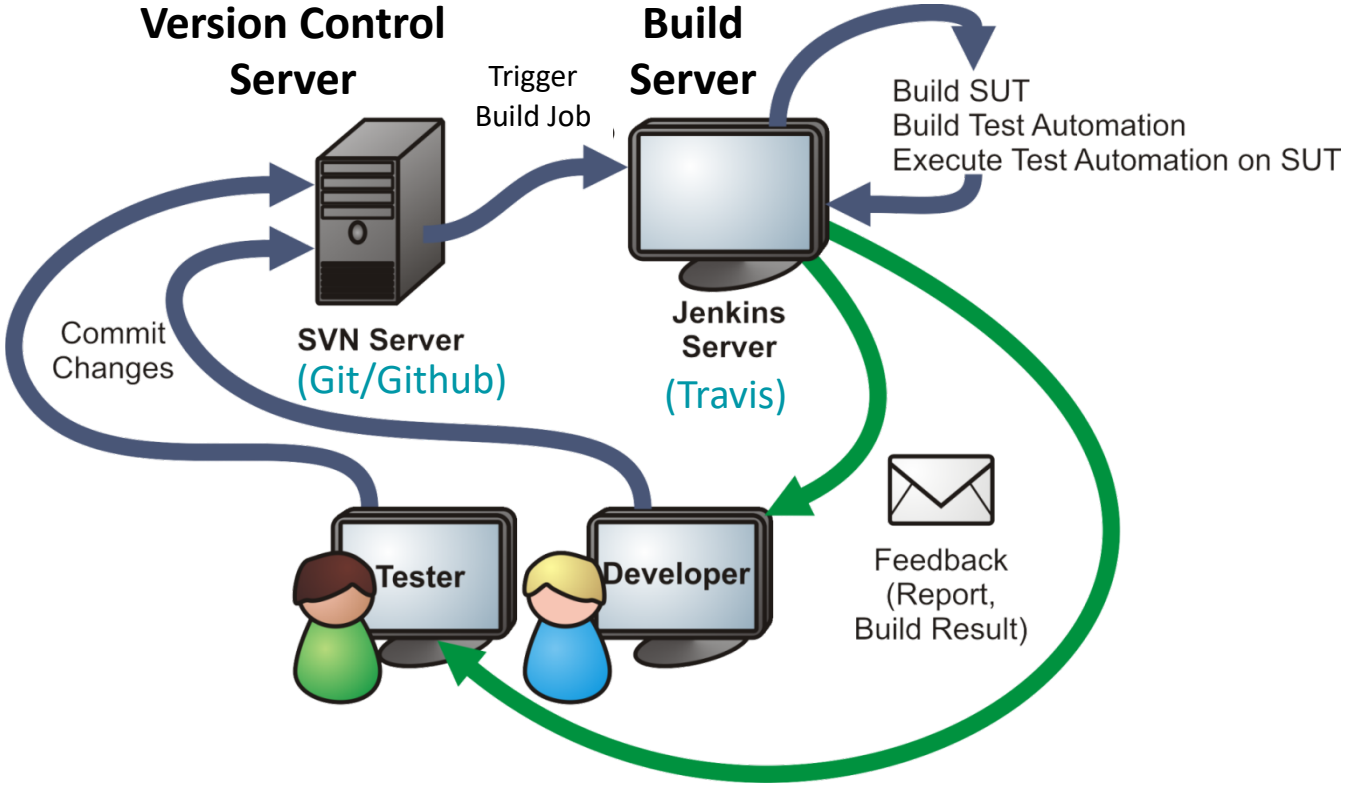
Code don’t compile => **YOU BROKE THE BUILD**

Run the automatic tests

Email build and test results to the team

E.g. Hudson, Jenkins, CruiseControl, Travis

Continuous Integration



You Broke the Build!


YOU BROKE THE BUILD!
This is Agnes and she's none too pleased with you, Bub. You broke the build. You should have known that you were making breaking changes, but you checked them in anyway.
If you start using continuous integration, Agnes won't have to come back.
www.YouBrokeTheBuild.com

Summary

Development environment overview and tools

Emulation != Simulation

App Workflow

1. Define resources & XML
2. Implement application classes in code (Java)
3. Package application
4. Install & run application

Version Control & Continuous Integration