Lecture 01: Touch & Input
IN721: Design and Development of Applications for Mobile Devices
Semester One, 2020

Kaiako: Grayson Orr

Te Kura Matatini ki Otago, Ōtepoti, Aotearoa

Wednesday, 19 February

Administration

- Click here to download the **course directive**
- Click here to view the **course materials repository**. Please clone this repository

Lecture 01: Touch & Input Topics

- Brief History
- Linux Kernel
- Software Stack
- Open-Source Community
- QEMU
- Kotlin
- Android Studio
- ArrayAdapters
- Inner class

Brief History

- Founded in Palo Alto, California in October 2003
- Early intentions were to develop an advanced operating system for digital cameras
- Google acquired Android Inc in July 2005
- Developed by a group of developers known as the Open Handset Alliance (OHA)
- De facto software for numerous smartphone manufacturing companies

Brief History

- Mascot of Android is a green robot
- Designed by Irina Blok on November 5, 2007 when Android was announced
- No official name, though the Android team at Google call it **Bugdroid**
- One of most recognisable icons in the technology world

Linux Kernel

- Based on the Linux kernel's LTS branches
- Android targets versions 4.14, 4.4 & 4.9 of the Linux kernel
- A Linux distribution according to the Linux Foundation

Software Stack

- Application
- Application framework
- Libraries
- Android runtime
- Hardware abstraction layer
- Linux kernel

Open-Source Community

- Android's source code is released by Google under an open-source license

Market Share

- Android - 74.3%
- iOS - 24.76%
- Reference - StatCounter (Mobile OS Market Share)

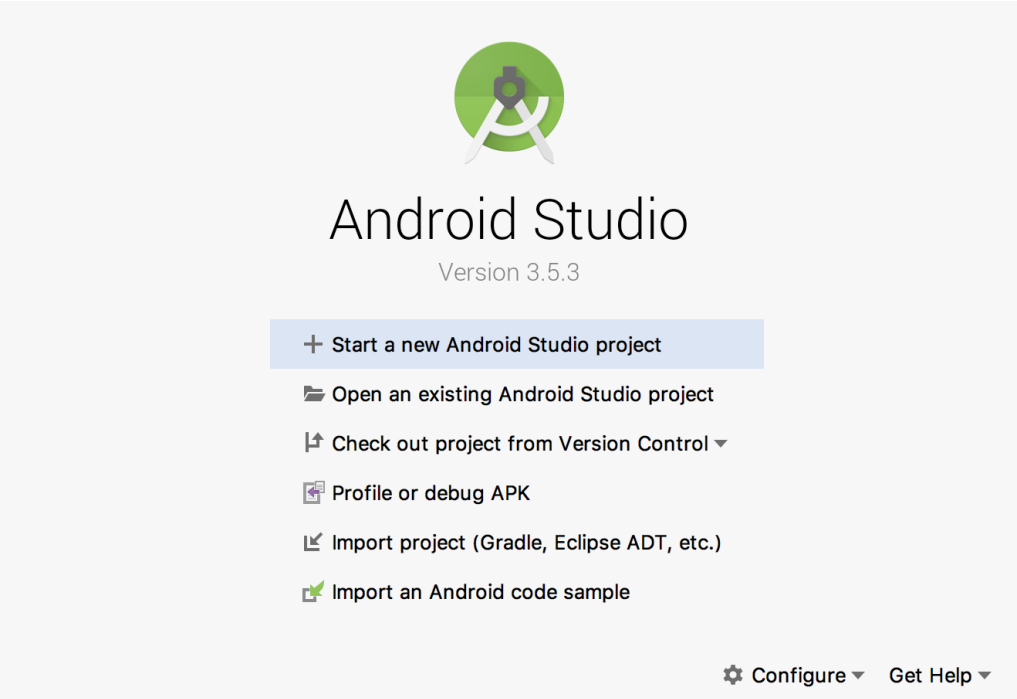
QEMU

- Quick EMUlator
- Open-source competitor to VMware Workstation, VirtualBox, HyperV
- Android emulator is built on top of the QEMU emulator

Kotlin

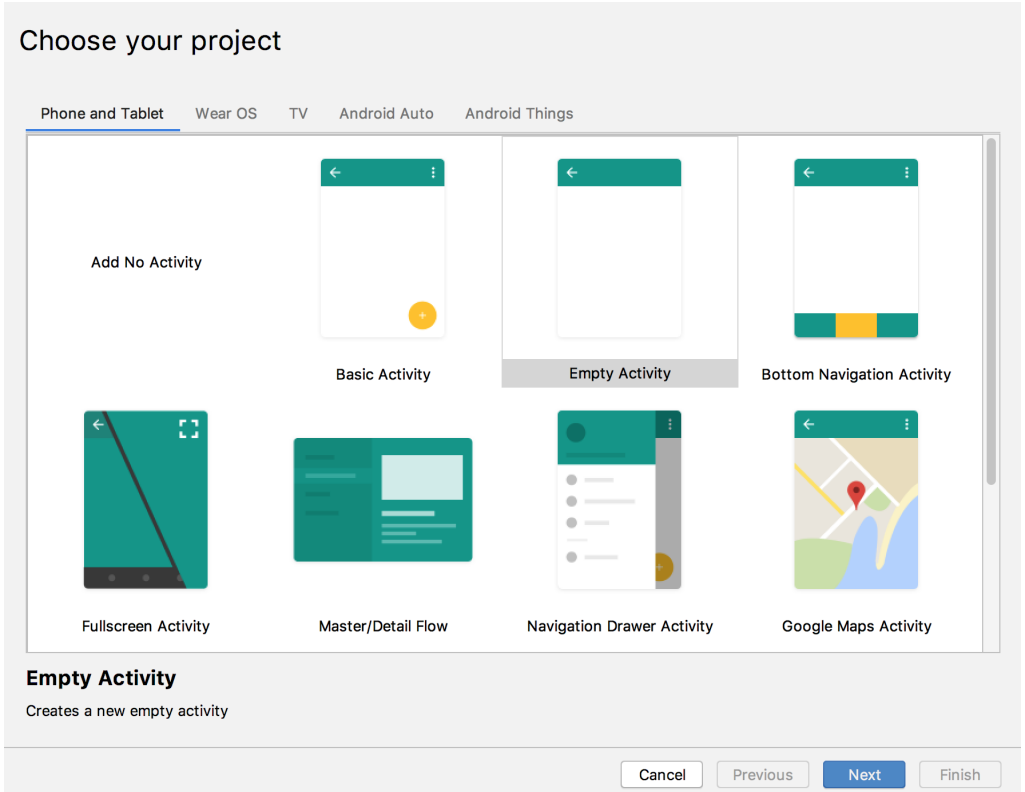
- Cross-platform
- Statically typed
- Type inference
- Interoperable with Java
- Preferred programming language for Android application developers
- One of my favourite programming languages

Android Studio: Application Setup



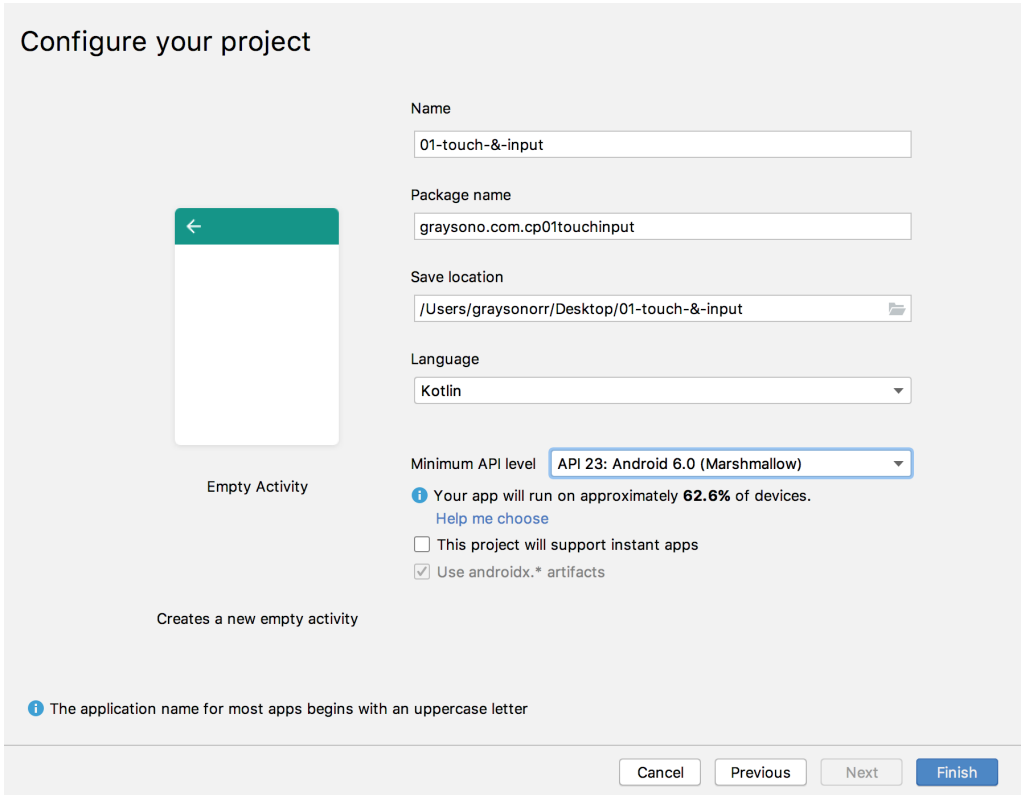
Android Studio: Application Setup

- Choosing a project
  - Basic activity
  - Empty activity
  - Bottom navigation activity



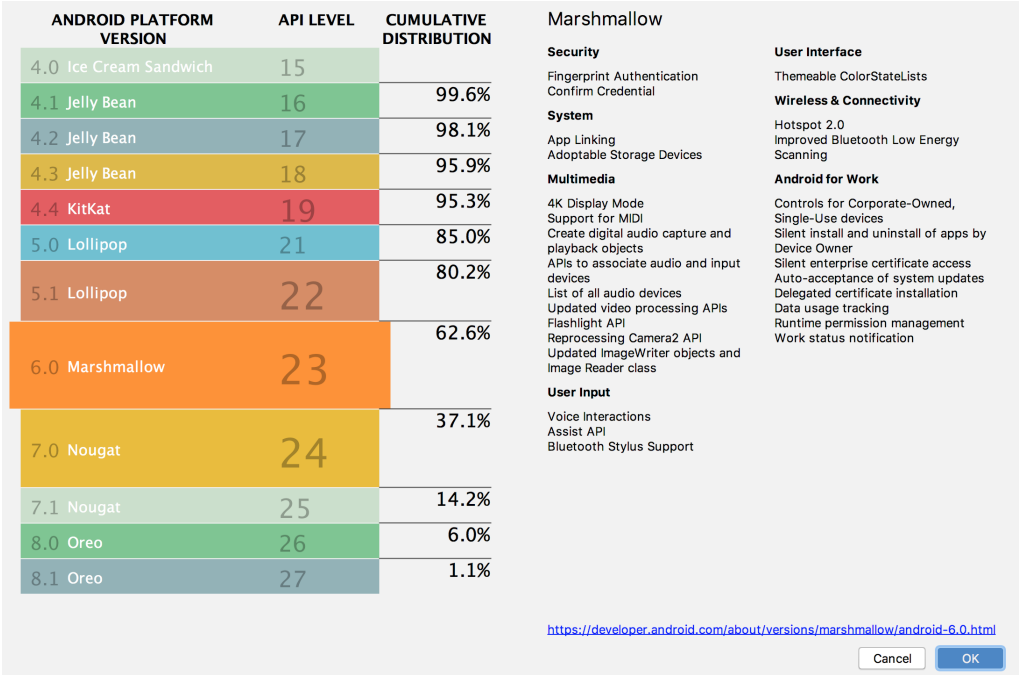
Android Studio: Application Setup

- Configuring a project
  - Name
  - Package name
  - Save location
  - Language
  - Minimum API level



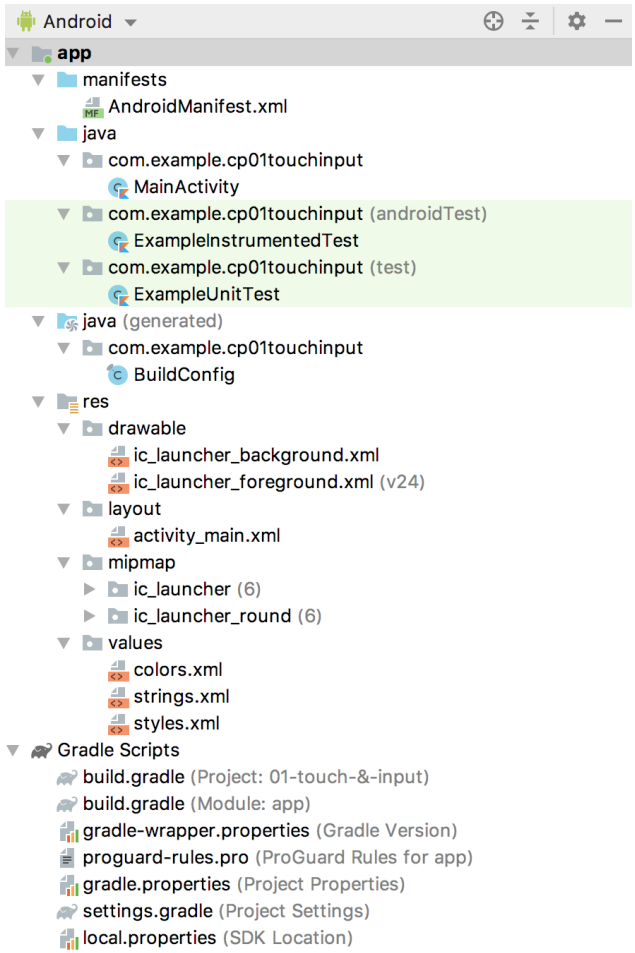
Android Studio: Application Setup

- Android platform version
- Cumulative distribution



Android Studio: Application Setup

- Directory structure



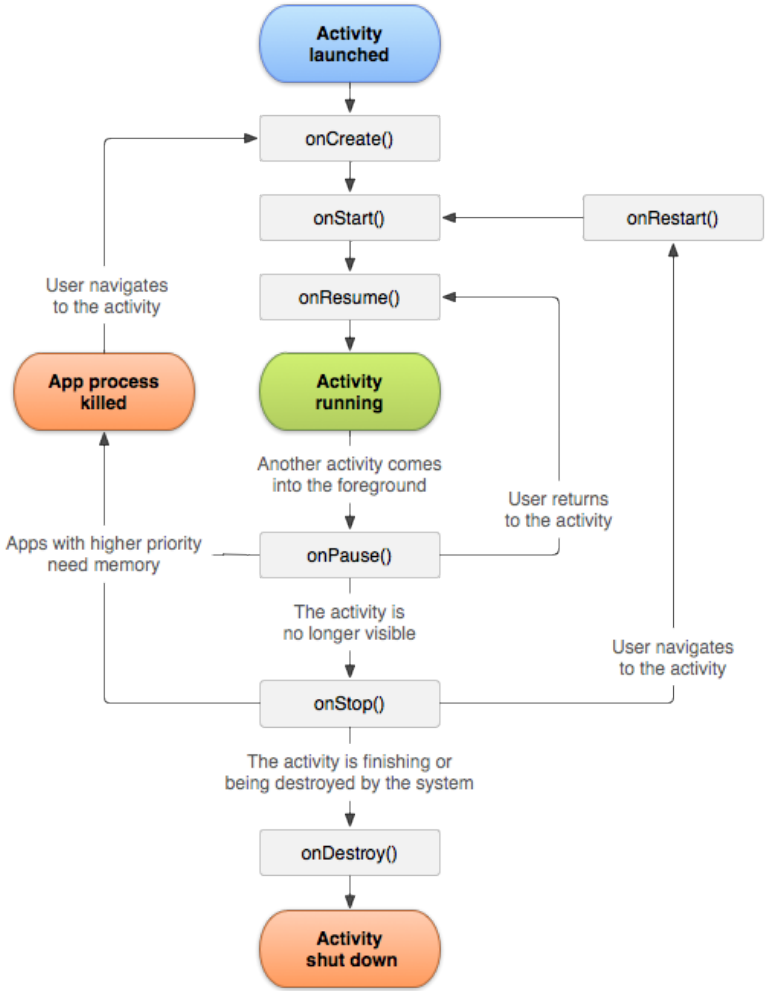
Android Studio: MainActivity.kt

- AppCompatActivity
- Activity lifecycle - onCreate()

MainActivity.kt

```
1  package com.example.cp01touchinput
2
3  import androidx.appcompat.app.AppCompatActivity
4  import android.os.Bundle
5
6  class MainActivity : AppCompatActivity() {
7
8      override fun onCreate(savedInstanceState: Bundle?) {
9          super.onCreate(savedInstanceState)
10         setContentView(R.layout.activity_main)
11     }
12 }
13
```

Android Studio: Activity Lifecycle



Inner Class

- Interface - OnClickListener
- android.view.View.OnClickListener
- A callback to be invoked when a view is clicked
- Public methods - onClick(v: View!): Unit

MainActivity.kt

```kotlin
package com.example.cp01touchinput

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import android.widget.Toast
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        btnClick.setOnClickListener(ButtonOnClickListener())
    }

    inner class ButtonOnClickListener : View.OnClickListener {
        override fun onClick(view: View?) {
            Toast.makeText( context: this@MainActivity,  text: "Click Me", Toast.LENGTH_LONG).show()
        }
    }
}
```

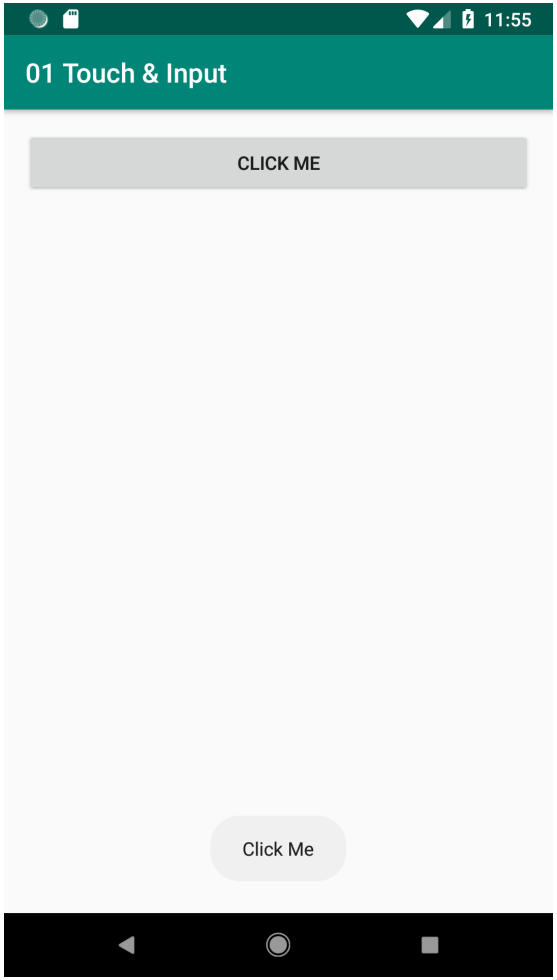
Alternative To Inner Class

- Simplified

MainActivity.kt

```kotlin
package com.example.cp01touchinput

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import android.widget.Toast
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        btnClick.setOnClickListener(View.OnClickListener { it: View!
            Toast.makeText( context: this@MainActivity,  text: "Click Me", Toast.LENGTH_LONG).show()
        })
    }
}
```

Emulator - OnClickListener



ARRAYADAPTER

- arrayOf - Kotlin

MainActivity.kt

```kotlin
package com.example.cp01touchinput

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.ArrayAdapter
import android.widget.Spinner
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        val daysOfWeek = arrayOf("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun")
        populateSpinner(spnDaysOfWeek, daysOfWeek)
    }

    private fun populateSpinner(spinner: Spinner, array: Array<String>) {
        val layoutID: Int = android.R.layout.simple_spinner_item
        spinner.adapter = ArrayAdapter( context: this@MainActivity, layoutID, array)
    }
}
```

Alternative To ArrayOf

- arrays.xml
- values > New > Values resource file

arrays.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="days_of_week">
        <item>Mon</item>
        <item>Tue</item>
        <item>Wed</item>
        <item>Thu</item>
        <item>Fri</item>
        <item>Sat</item>
        <item>Sun</item>
    </string-array>
</resources>
```

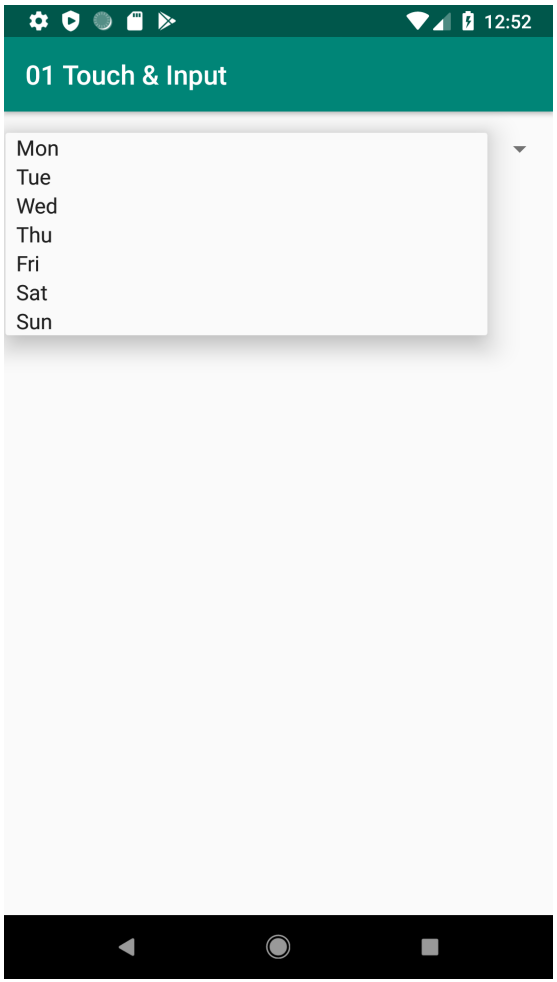
Alternative To ArrayOf

- resources.getStringArray(id: Int)

MainActivity.kt

```kotlin
package com.example.cp01touchinput

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.ArrayAdapter
import android.widget.Spinner
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        val daysOfWeek = resources.getStringArray(R.array.days_of_week)
        populateSpinner(spnDaysOfWeek, daysOfWeek)
    }

    private fun populateSpinner(spinner: Spinner, array: Array<String>) {
        val layoutID: Int = android.R.layout.simple_spinner_item
        spinner.adapter = ArrayAdapter( context: this@MainActivity, layoutID, array)
    }
}
```

Emulator - ArrayAdapter



Android Studio: AndroidManifest.xml

- Describes essential information about an application to the Android build tools, Android OS & Google Play
- Manifest file is required to declare:
  - Application's package name
  - Components of the application
  - Permissions
  - Hardware & software features

AndroidManifest.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      package="com.example.cp01touchinput">
4
5      <application
6          android:allowBackup="true"
7          android:icon="@mipmap/ic_launcher"
8          android:label="01-touch-&-input"
9          android:roundIcon="@mipmap/ic_launcher_round"
10         android:supportsRtl="true"
11         android:theme="@style/AppTheme">
12         <activity android:name=".MainActivity">
13             <intent-filter>
14                 <action android:name="android.intent.action.MAIN" />
15
16                 <category android:name="android.intent.category.LAUNCHER" />
17             </intent-filter>
18         </activity>
19     </application>
20
21 </manifest>
```

Android Studio: Resources

- Android resource directories
  - Drawable
  - Layout
  - Mipmap
  - Values

Android Studio: Resources - Drawable

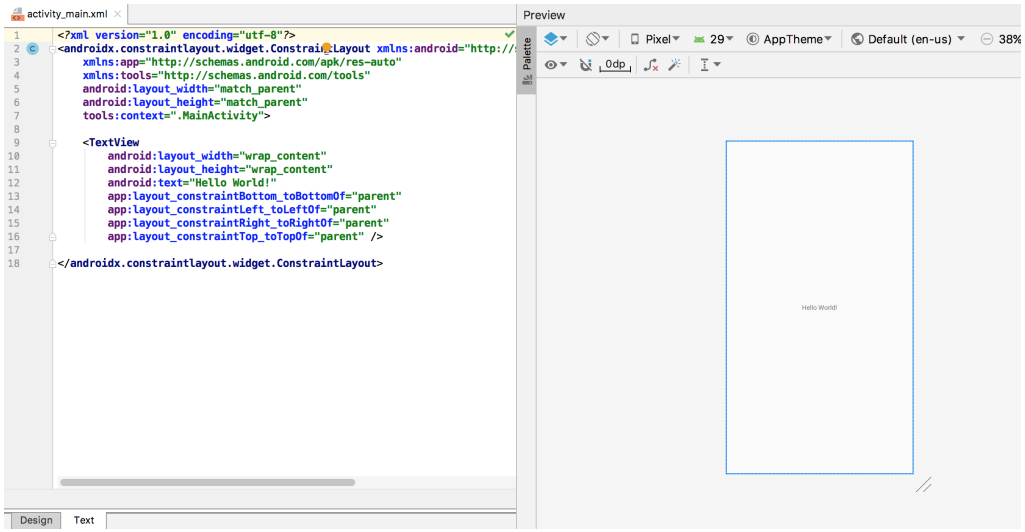
- General abstraction for something that can be drawn
- Type of resource retrieved for drawing things to the screen
- Generic API for dealing with an underlying visual resource
- Examples of drawables - bitmaps & vectors

Android Studio: Resources - Layout

- Defines the structure for a user interface in your application
- All elements in the layout are built using a hierarchy of **View** & **ViewGroup** objects
- **View** objects are called widgets
- **ViewGroup** objects are called layouts
- Two ways to declare a layout:
  - Declare user interface elements in XML - activity_main.xml
  - Instantiate layout elements at runtime

Android Studio: Resources - Layout

- Each layout file must contain exactly one root element
  - Must be a **View** or **ViewGroup** object
- Add additional layout objects or widgets as child elements



Android Studio: Resources - Mipmap

- Drawable files for different launcher icon densities
- Image Asset Studio
- Generate application icons from material icons, custom images & text strings
- Android Asset Studio - Roman Nurik

Android Studio: Resources - Values

- XML files that contain simple values such as colors, strings & styles
- Each child of the <resources> element defines a single resource
- For example, a <string> element creates an R.string resource

Android Studio: Resource Types

- Menu & font - lecture 03
- XML - lecture 08
- Mipmap - lecture 14
- Anim - lecture 26

Android Studio: Gradle

- build.gradle

app

Configure project in Project Structure dialog. Open Project Structur

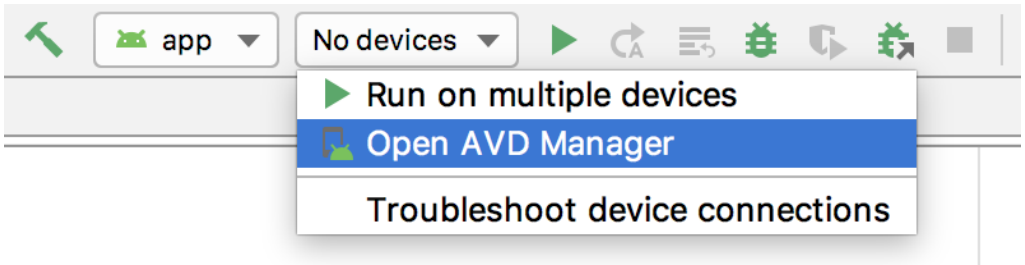
```
apply plugin: 'com.android.application'

apply plugin: 'kotlin-android'

apply plugin: 'kotlin-android-extensions'

android {
    compileSdkVersion 29
    buildToolsVersion "29.0.3"
    defaultConfig {
        applicationId "com.example.cp01touchinput"
        minSdkVersion 23
        targetSdkVersion 29
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation"org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
    implementation 'androidx.appcompat:appcompat:1.0.2'
    implementation 'androidx.core:core-ktx:1.0.2'
    implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'androidx.test.ext:junit:1.1.0'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.1.1'
}

```

Android Studio: Emulator

- Simulates Android devices on your computer
- Test an application on a variety of devices & API levels
- Provides almost all of the capabilities of a real Android device

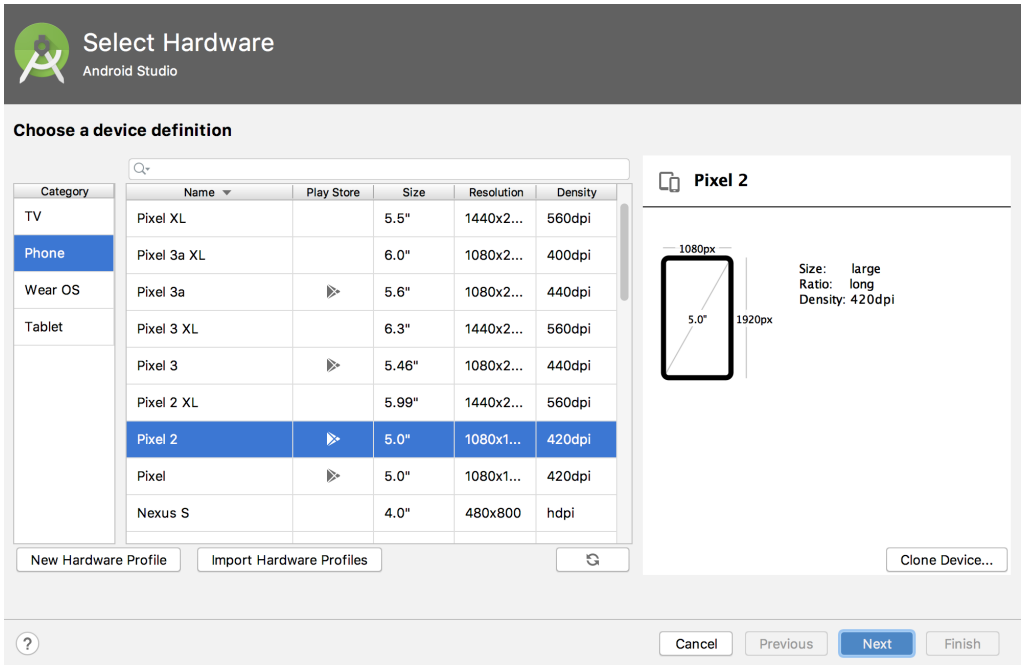
Android Studio: Emulator

- AVD Manager



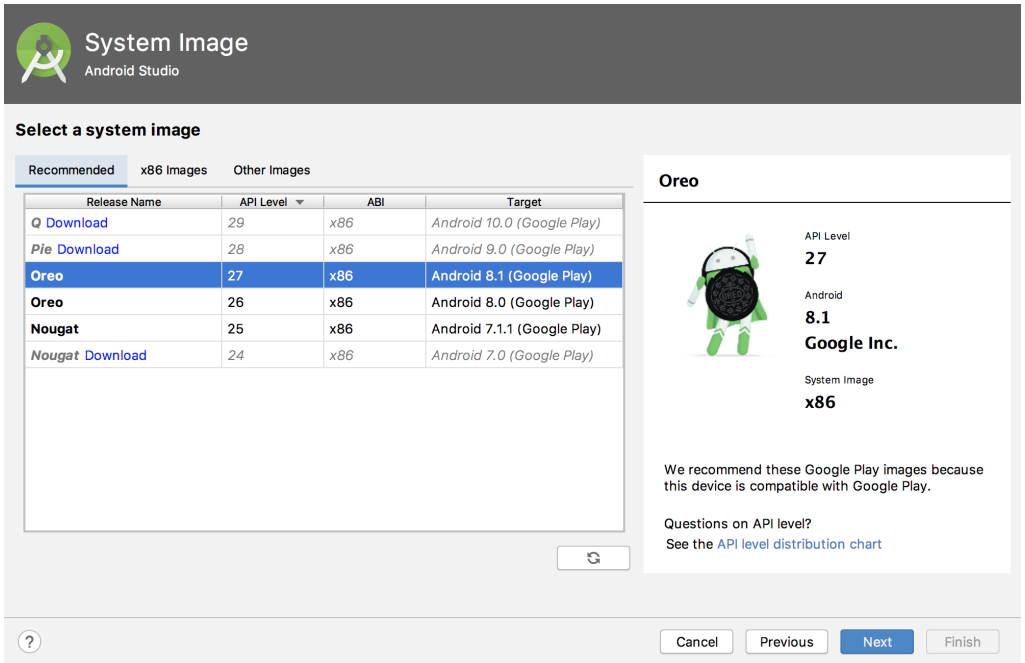
Android Studio: Emulator

- Selecting hardware
  - Category
  - Name



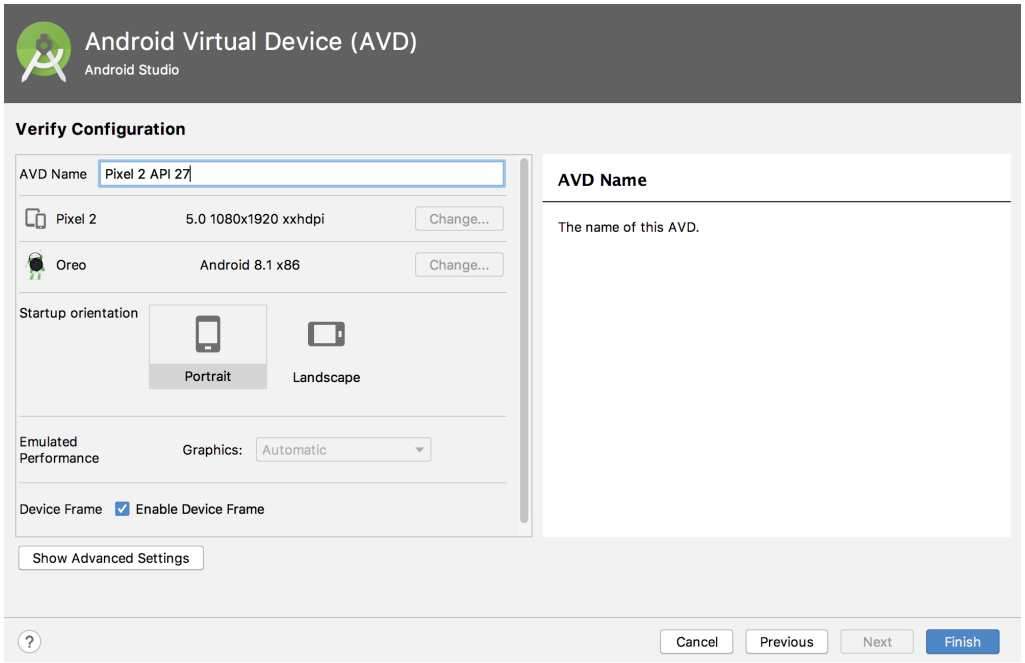
Android Studio: Emulator

- System Image
  - Release name
  - API level

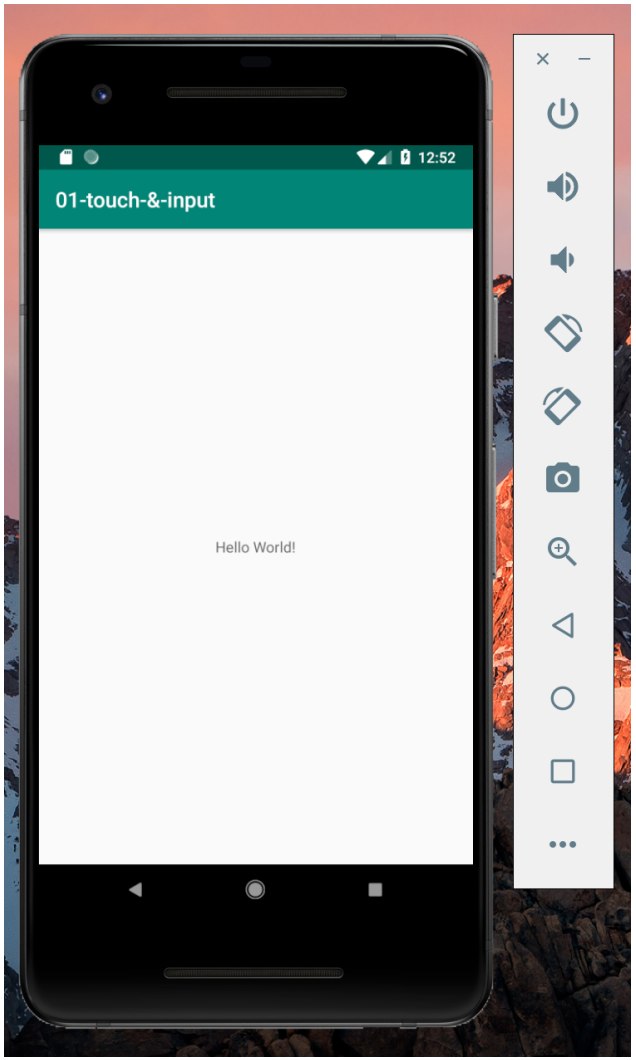


Android Studio: Emulator

- Verifying configuration
  - AVD name
  - Startup orientation



Android Studio: Emulator



Practical

- Series of tasks covering today's lecture
- Worth 1% of your final mark for the Design and Development of Applications for Mobile Devices course
- Deadline: Friday, 12 June at 5pm

Assessment 1 & 2

- Two assessments worth worth 20% & 25% - one individual & one group
- Worth 45% of your final mark for the Design and Development of Applications for Mobile Devices course
- Submission via **Assignments** tab on Microsoft Teams & GitHub
- Deadline: refer to course directive

Exams

- Five individual exams worth 6% each
- Worth 30% of your final mark for the Design and Development of Applications for Mobile Devices course
- Submission via **Assignments** tab on Microsoft Teams
- Deadline: refer to course directive

GitHub Repositories

- A1: Language Translator - http://bit.ly/mobile-language-translator
- A2: Wishlist - http://bit.ly/mobile-wishlist
- Practicals - http://bit.ly/mobile-practicals
- Click here to view the **GitHub Classroom Setup** video

Learner Capability Framework

- LCF is based on national & international research
- **iamcapable** - web-based tool
  - Track the development of learner capabilities
  - Produce verified evidence of these capabilities
- Come see me for more information

Formative Assessment

- Formative assessment questions:
  - What are the fourAndroid resource directories?
  - What does the onCreate() method do in the Android activity lifecycle?
- Deadline: Friday, 21 February at 8am
- Click here to fill out the formative assessment